SEQUENTIAL DECISION MAKING WITH RESOURCE CONSTRAINTS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Ashwinkumar Badanidiyuru Varadaraja

August 2014

SEQUENTIAL DECISION MAKING WITH RESOURCE CONSTRAINTS

Ashwinkumar Badanidiyuru Varadaraja, Ph.D.

Cornell University 2014

In sequential decision making, an algorithm interacts with an environment, where it can learn from the feedback of its past actions. A model for sequential decision making with partial feedback is the multi-armed bandit problem. This model has also found applications to a very diverse set of problems such as sequential design of experiments including medical decision-making, learning click-through rates in search engines, economic theory, network routing, etc.

We study a fundamental feature in many of these applications, which is the presence of one or more limited-supply resources that are consumed during the decision process. Existing literature lacked general models for this feature and offered very limited treatment of such problems. We propose models which capture many of these applications and give tight performance guarantees.

BIOGRAPHICAL SKETCH

Ashwinkumar Badanidiyuru Varadaraja was born in Udupi, Karnataka, India on Feb 17, 1985 and grew up in Davangere, Karnataka. He spent four exciting years studying computer science at IIT Madras and got his B.Tech in 2008. Then he moved to Ithaca where he expects to get a PhD in Computer Science with a minor in Applied Math from Cornell University in August 2014.

This document is dedicated to my wonderful parents.

Dr. Shakuntala V. Rao and Dr. B. A. Varadaraja Rao.

ACKNOWLEDGEMENTS

I still remember the day I got an offer for pursing PhD at Cornell and it was one of my happiest moments. The whole process during the last six years has been one of learning, meeting exciting people and of course minor roller coasters. Among these the one person who has shaped my thinking the most has been my advisor Bobby Kleinberg. Although I came to Cornell with interest in pursuing theoretical computer science, I must say that his course on Algorithms during my first semester was what cemented my passion towards it. Our weekly meetings were no less interesting, and I would be shocked at how he could help me in that little period of time and in many cases generate very useful ideas on the spot. His approach to research along with the freedom he gave me is what made enjoy my research so much. I thank him heartily for all the guidance he gave me as an advisor and friend, and allowed me to become who I am.

There are many researchers who have significantly influenced both my research tastes and thinking. I would like to thank all my research mentors and collaborators including Alex Slivkins, Jan Vondrák, Yaron Singer, Shahar Dobzinski, Hu Fu, Lior Seeman, Alekh Agarwal, Miroslav Dudík, Robert E. Schapire, John Langford, Amin Karbasi, and Andreas Krause. A special thanks to Alex Slivkins for hosting me for two internships and collaborating with me on what has become two major chapters in this thesis. A special thanks also to Jan Vondràk for also hosting me for an internship and allowing me to collaborate on submodular optimization.

I would also like to thank my committee members Johannes Gehrke and Éva Tardos for giving me very good advice. A special thanks to Johannes for guiding me in my second year when I was still searching for a thesis topic and

allowing me to work under him on Data Privacy. I also thank other faculty members at Cornell including Thorsten Joachims, Jon Kleinberg, Dexter Kozen, David Willamson, David Shmoys, Aaron Wagner, Mark Lewis and Kenneth S. Brown for giving timely advice and for teaching very exciting courses.

A huge part of what kept me grounded during my years in Ithaca was many interactions with many good friends including Renato Paes Leme, Hu Fu, Hussam Abu-Libdeh, Varun Hiremath, Vishal Chandrashekar, Sumit Kumar, Karthik Raman, Adith Swaminathan, Anshumali Shrivastava. I also want to thank many of my pre-Ithaca friends Bandi, Naresh, Balu, George, Kurma, Nagarvind, Putta, Chethan, Babu whom I kept bumping to either in conferences or other locations. A special thanks to Naresh and his room mate Vijay, Bandi, George, Nagarvind and his room mate Nagendra who allowed me to hitch a stay when I was visiting.

One addition to my family during the last semester has been my wife Shobha. I couldn't have imagined I would connect with someone so much in such a short period of time. I thank her for her encouragement and for the lovely time we are having together.

Last, but not the least I thank my wonderful parents, my amma Dr. Shakuntala V. Rao and my papa Dr. B. A. Varadaraja Rao for their unconditional love and support before and through my PhD.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction

How should a decision maker manage his/her resources when making decisions in a sequential manner? The focus of this thesis is in addressing this challenge. In sequential decision making, an agent interacts with the environment in a sequential manner. While the environment is assumed to behave in a certain way, the agent lacks some of the information about other aspects of the environment. This uncertainty is the main source of difficulty and the agent strives to learn the missing information based on the feedback of its past actions.

For more than fifty years, the multi-armed bandit problem (henceforth, *MAB*) has been one of the predominant theoretical models for sequential decision making. It embodies the tension between exploration and exploitation, "the conflict between taking actions which yield immediate reward and taking actions whose benefit (e.g. acquiring information or preparing the ground) will come only later," to quote Whittle's apt summary [54]. Owing to the universal nature of this conflict, it is not surprising that MAB algorithms have found diverse applications ranging from medical trials, to communication networks, to Web search and advertising.

A common feature in many of these application domains is the presence of one or more limited-supply resources that are consumed during the decision process. For example, scientists experimenting with alternative medical treatments may be limited not only by the number of patients participating in the

study but also by the cost of materials used in the treatments. A website experimenting with displaying advertisements is constrained not only by the number of users who visit the site but by the advertisers' budgets. A retailer engaging in price experimentation faces inventory limits along with a limited number of consumers. The literature on MAB problems lacks general models that encompass these sorts of decision problems with supply limits. In this thesis we introduce and study various models to address this issue and present algorithms with provable performance guarantees. In fact, we prove that for many of the addressed situations the algorithms have provably optimal performance guarantees.

The rest of the introduction is organized as follows. In section 1.2 we give the motivating examples for the problems we study. Then, in section 1.3 we give the formal models which model these problems. In section 1.4 we note the related work, following which we state our contributions in 1.5. Finally, we end the section by an outline of the thesis in 1.6 along with bibliographic notes in 1.7.

1.2 Motivating Examples

The presence of resources which can be consumed in a limited quantity is quite common in many applications. Hence, it is no surprise that a lot of problems fall under the umbrella of sequential decision making with resource constraints. We list a couple of such examples below.

Dynamic pricing. Consider an airline which is interested in selling a limited number of seats on a flight. The airline wants to maximize its revenue and is

offering tickets on a website (e.g. expedia). Buyers appear one by one and are interested in purchasing a ticket if the price quoted to them is smaller than their value for the ticket. The airline presents a take-it-or-leave-it price to each buyer. Additionally, the airline can update its price dynamically to take into account the number of seats left, number of buyers who are yet to arrive, and the history of transactions so far.

While the basic version of dynamic pricing is a stylized model, we will incorporate many extensions of the problem as special cases of the problem that we study. These include buyers interested in more than one item (non-unit demand), limited inventory of multiple types of products, multiple products made from limited inventory of common raw materials, bundling, and buyer targeting. We will make all of these models formal as special cases of two very general models that we consider in sections 3.7 and 4.7.

Dynamic procurement. The dynamic procurement problem is the dual of the dynamic pricing problem, where there is one buyer of goods and multiple sellers appear one by one. Consider the example where a requester (buyer of services) wants to spend at most B dollars to get as many images labelled as possible. To achieve it the requester uses Amazon's mechanical turk platform, in which workers (sellers of services) arrive one by one and are offered a take-it-or-leave-it price for doing the work. The requester can adjust the price dynamically to take into account the money spent so far, number of workers who are yet to arrive, and the history of transactions to far.

Similar to dynamic pricing we will also consider other variants of dynamic procurement. These include workers who can do more than one job (non-unit supply), multiple-types of jobs, complicated menus of jobs, etc. We will make

all of these models formal as special cases of two very general models that we consider in sections 3.7 and 4.7.

Pay-per click ads. Consider a search engine such as Google which wants to learn the click through rates of advertisements. But, the search engine is constrained to not charge each advertiser more than the budget constraint set by the advertiser. Here the search engine can show advertisements adaptively based on the budget remaining for each advertiser, the number of queries yet to appear, and the history of clicks for each advertisement.

Other applications. Along with the above three examples we will also consider many other applications in section 3.7. These include setting reserve prices in repeated auctions with limited inventory, adjusting a routing protocol with limited bandwidth resources, and adjusting a scheduling policy with limited machine resources.

All the above examples have many variants which are interesting. While the basic case is interesting in itself, when applying models to answer queries over a web server one has to consider cookie data of users to get targeted results. We consider the applications without the contextual data in section 3.7 and with the contextual data in 4.7. We will also consider a special case of these applications in chapter 5 where we can consider more general models of utility, where the total utility of the algorithm is not a sum of the utilities in each round, but a more complicated function.

1.3 Problem formulation

1.3.1 Existing models

We will first define the existing models in the literature on which we will build upon. We will discuss the relevant literature along with related models in the next section 1.4.

Definition 1.3.1 (Stochastic multi-armed bandit)**.** There are a fixed and known, finite set of *m arms* (possible actions), denoted A. In each round t, an algorithm ALG picks an arm $a_t \in A$ and receives reward $r_t \in [0, 1]$. The reward r_t is revealed to the algorithm after the round and is assumed to be drawn from an *unknown* distribution $D(a_t)$. The goal of the algorithm is to maximize the expected total reward. The benchmark to evaluate any algorithm ALG at any time step T will be comparing the reward of ALG to what is the best achievable reward if the distributions $\{D(a)|a \in A\}$ are known, i.e. comparing it to $T \cdot \max_{a \in A} \mathbb{E}[r(a)]$.

Definition 1.3.2 (Contextual stochastic bandit)**.** There are a fixed and known, finite set of *m arms* (possible actions), denoted A. In each round t, a context x_t is picked at random from distribution $\mathcal{D}_X$ over possibly infinite known set of possible contexts X. Then an algorithm ALG observes a context x_t and then picks an arm $a_t \in A$. After picking the arm ALG receives reward $r_t \in [0, 1]$. The reward r_t is revealed to the algorithm after the round and is assumed to be drawn from an *unknown* distribution $D(x_t, a_t)$. The goal of the algorithm is to maximize the expected total reward.

ALG also has access to a finite set Π of *policies* – mappings from contexts to actions. The benchmark to evaluate any algorithm ALG at any time step T will be

comparing the reward of ALG to the maximum achievable reward by following the recommendations of a fixed policy $\pi \in \Pi$ from beginning to time step T, i.e. comparing it to $T \cdot \max_{\pi \in \Pi} \mathbb{E}_{x \sim \mathcal{D}_X}[r(x, \pi(x))]$.

While the input to the above two models is chosen stochastically, in section 5 we will also consider an input model where the complete input is chosen by an adversary, but the input appears in a random order.

1.3.2 Performance guarantees

For different problems we use different benchmarks. Consider any algorithm ALG which has reward r_t at time t, then to show performance guarantees we define the following two metrics.

Definition 1.3.3 (Regret)**.** An algorithm ALG has a regret γ with respect to benchmark BENCH if the expected reward of ALG differs from BENCH by at most γ, i.e

$$\texttt{BENCH} - \mathbb{E}\left[\sum_{i=1}^{T} r_t\right] \leq \gamma$$

Definition 1.3.4 (Competitive ratio)**.** An algorithm ALG has a competitive ratio γ with respect to benchmark BENCH if the expected reward of ALG is at least a γ factor of BENCH, i.e

$$\frac{\texttt{BENCH}}{\mathbb{E}\left[\sum_{i=1}^{T} r_t\right]} \leq \gamma$$

1.3.3 Bandits with Knapsacks

Henceforth we will call this model as BwK.

Problem formulation. There is a fixed and known, finite set of *m arms* (possible actions), denoted A. There are d resources being consumed. In each round t, an algorithm picks an arm $a_t \in A$, receives reward $r_t \in [0,1]$, and consumes some amount $c_{t,i} \in [0,1]$ of each resource i. The values r_t and $c_{t,i}$ are revealed to the algorithm after the round. There is a hard constraint $B_i \in \mathbb{R}_+$ on the consumption of each resource i; we call it a *budget* for resource i. The algorithm stops at the earliest time τ when one or more budget constraint is violated; its total reward is equal to the sum of the rewards in all rounds strictly preceding τ. The goal of the algorithm is to maximize the expected total reward.

The vector $(r_t; c_{t,1}, c_{t,2}, \ldots, c_{t,d}) \in [0,1]^{d+1}$ is called the *outcome vector* for round t. We assume *stochastic outcomes*: if an algorithm picks arm a, the outcome vector is chosen independently from some fixed distribution $D(a)$ over $[0,1]^{d+1}$. The distributions $D(a)$, $a \in A$ are latent. The tuple $(D(a) : a \in A)$ comprises all latent information in the problem instance. A particular `BwK` setting (such as "dynamic pricing with limited supply") is defined by the set of all feasible tuples $(D(a) : a \in A)$. This set, called the *BwK domain*, is known to the algorithm.

We will assume that there is a fixed *time horizon* T, known in advance to the algorithm, such that the process is guaranteed to stop after at most T rounds. One way of assuring this is to assume that there is a specific resource, say resource 1, such that every arm deterministically consumes B_1/T units whenever it is picked. We make this assumption henceforth. Without loss of generality, $B_i \leq T$ for every resource i.

For technical convenience, we assume there exists a *null arm*: an arm with 0 reward and 0 consumption. Equivalently, an algorithm is allowed to spend a unit of time without doing anything.

Benchmark. We compare the performance of our algorithms to the expected total reward of the optimal dynamic policy given all the latent information, which we denote by OPT. Note that OPT depends on the latent structure μ, and therefore is a latent quantity itself. Time-invariant policies — those which use the same distribution $\mathcal{D}$ over arms in all rounds — will also be relevant to the analysis of one of our algorithms. Let $\texttt{REW}(\mathcal{D}, \mu)$ denote the expected total reward of the time-invariant policy that uses distribution $\mathcal{D}$.

Uniform budgets. We say that the budgets are *uniform* if $B_i = B$ for each resource i. Any BwK instance can be reduced to one with uniform budgets by dividing all consumption values for every resource i by B_i/B, where $B = \min_i B_i$. (That is tantamount to changing the units in which we measure consumption of resource i.) Our technical results are for BwK with uniform budgets. We will assume uniform budgets B from here on.

Useful notation. Let $\mu_a = \mathbb{E}_{\lambda_x \sim \mathcal{D}(a)}[\lambda_a] \in [0,1]^{d+1}$ be the expected outcome vector for each arm a, and denote $\mu = (\mu_a : a \in A)$. We call μ the *latent structure* of a problem instance. The BwK domain induces a set of feasible latent structures, which we denote $\mathcal{M}_{\texttt{feas}}$.

For notational convenience, we will write $\mu_a = (\, r(a,\mu);\ c_1(a,\mu), \ \ldots\ , c_d(a,\mu)\,)$. Also, we will write the expected consumption as a vector $c(a,\mu) = (\, c_1(a,\mu), \ \ldots\ , c_d(a,\mu)\,)$.

If $\mathcal{D}$ is a distribution over arms, let $r(\mathcal{D},\mu) = \sum_{a\in A} \mathcal{D}(a)\, r(a,\mu)$ and $c(\mathcal{D},\mu) = \sum_{a\in A} \mathcal{D}(a)\, c(a,\mu)$ be, respectively, the expected reward and expected resource consumption in a single round if an arm is sampled from distribution $\mathcal{D}$.

When discussing one of our algorithm, it will be useful to represent the latent

values and the algorithm's decisions as matrices and vectors. For this purpose, we will number the arms as $a_1, \ldots, a_m$ and let $r \in \mathbb{R}^m$ denote the vector whose j^{th} component is $r(a_j, \mu)$. Similarly we will let $C \in \mathbb{R}^{d \times m}$ denote the matrix whose $(i, j)^{\text{th}}$ entry is $c_i(a_j, \mu)$.

1.3.4 Resourceful contextual bandits

Henceforth we will call this model as RCB. We consider an online setting where in each round an algorithm observes a context x from a possibly infinite known set of possible contexts X and chooses an action a from a finite known set A. The world then specifies a reward $r \in [0, 1]$ and the resource consumption. There are d resources that can be consumed, and the resource consumption is specified by numbers $c_i \in [0, 1]$ for each resource i. Thus, the world specifies the vector $(r; c_1, \ldots, c_d)$, which we call the *outcome vector*; this vector can depend on the the chosen action a and the round. There is a known hard constraint $B_i \in \mathbb{R}_+$ on the consumption of each resource i; we call it a *budget* for resource i. The algorithm stops at the earliest time τ when any budget constraint is violated; its total reward is the sum of the rewards in all rounds strictly preceding τ. The goal of the algorithm is to maximize the expected total reward.

We are only interested in regret at a specific time T (*time horizon*) which is known to the algorithm. Formally, we model time as a specific resource with budget T and a deterministic consumption of 1 for every action. W.l.o.g., $B_i \leq T$ for every resource i.

For technical convenience, we assume there exists a *null action*: an action with 0 reward and 0 consumption except for time. That is, an algorithm can

spend one round without doing anything.[1]

Stochastic assumptions. We assume that there exists an unknown distribution $D(x, \vec{r}, \vec{c}_i)$, called the *outcome distribution*, from which each round's observations are created independently and identically, where the vectors are indexed by individual actions. In particular, context x is drawn from the marginal distribution $\mathcal{D}_{\mathtt{X}}(\cdot)$, and the observed reward and resource consumptions for each action a are drawn from the conditional distribution $D(\vec{r}_a, \vec{c}_{ia}|x)$. For clarity, we assume that the marginal distribution over contexts $D(x)$ is known.

Policy sets and the benchmark. An algorithm is given a finite set Π of *policies* – mappings from contexts to actions. Our benchmark is a hypothetical algorithm that knows the outcome distribution D, and makes optimal decisions given this knowledge. The benchmark is restricted to policies in Π: before each round, it must commit to some policy $\pi \in \Pi$, and then choose action $\pi(x)$ upon arrival of any given context x. The expected total reward of the benchmark is denoted $\texttt{OPT}(\Pi)$. *Regret* of an algorithm is $\texttt{OPT}(\Pi)$ minus the algorithm's expected total reward.

Uniform budgets. We say that the budgets are *uniform* if $B_i = B$ for each resource i. Any problem instance can be reduced to one with uniform budgets by dividing all consumption values for every resource i by B_i/B, where $B = \min_i B_i$. (That is tantamount to changing the units in which we measure consumption of resource i.) We assume uniform budgets B from here on.

Notation. Let $r(\pi) = E_{(x,\vec{r})\sim D}[\vec{r}_{\pi(x)}]$ and $c_i(\pi) = E_{(x,\vec{c}_i)\sim D}[\vec{c}_{i\pi(x)}]$ be the expected per-

[1]The null action is needed to ensure the existence of an "LP-perfect distribution" (defined in Section 4.1.1).

round reward and the expected per-round consumption of resource i for policy π. Similarly, define $r(P) = E_{\pi\sim P}[r(\pi)]$ and $c_i(P) = E_{\pi\sim P}[c_i(\pi)]$ as the natural extension to a distribution P over policies.

The tuple $\mu = (\,(r(\pi); c_1(\pi)\,, \ \ldots\ , c_d(\pi)) : \ \pi \in \Pi\,)$ is called the *expected-outcomes tuple*.

For a distribution P over policies, let $P(\pi)$ is the probability that P places over policy π. By a slight abuse of notation, let $P(a|x) = \sum_{\pi(x)=a} P(\pi)$ be the probability that P places on action a given context x. Thus, each context x induces a distribution $P(\cdot|x)$ over actions.

1.3.5 Dynamic procurement

In Chapter 5 we also consider a special case of `BwK` (specifically Dynamic procurement) and its generalizations. In this model there are n agents, denoted $\mathcal{N}$, each offering a service for which they associate a private cost $c_i \in \mathbb{R}_+$. The buyer has a public (known to the mechanism designer and maybe known to all agents) budget $B \in \mathbb{R}_+$ and a public nondecreasing utility function $f : 2^{\mathcal{N}} \to \mathbb{R}_+$ over the subsets of agents. The sellers arrive one by one and are offered a take-it-or-leave-it price. The seller sells the item iff the price offered is above the seller's cost, otherwise he/she leaves forever. We consider the following two special cases.

1. The function f is a normalized monotone positive submodular function and the function f and costs c_i are adversarially chosen. Further the agents appear in random order. A set function f is said to be normalized mono-

tone positive submodular if it satisfies the following four properties.

(a) It is normalized if $f(\emptyset) = 0$

(b) It is positive if for each set $S \subseteq \mathcal{N}$ it satisfies $f(S) \geq 0$.

(c) It is monotone if for each pair of sets $S \subseteq T \subseteq \mathcal{N}$ it satisfies $f(S) \leq f(T)$.

(d) It is submodular if for each pair of sets $S, T \subseteq \mathcal{N}$ and for any element $e \in \mathcal{N}$ with $e \notin T$ it satisfies $f(S \cup \{e\}) - f(S) \geq f(T \cup \{e\}) - f(T)$.

In the rest of the thesis we will overload the notation and call any normalized monotone positive submodular function as a submodular function. We will also call this case as a submodular procurement market.

2. The function f is a symmetric submodular function and the costs c_i are drawn i.i.d from an unknown distribution. A set function f is said to be symmetric submodular function if it is a submodular function and it satisfies the following property.

   - For any pair of sets $S, T \subseteq \mathcal{N}$ with $|S| = |T|$ we have $f(S) = f(T)$

   We will call this case as a symmetric submodular procurement market.

1.4 Related work

The work on sequential decision making with resource constraints is at the intersection of many well studied problems and builds on a lot of existing work which we discuss below.

Multi-armed bandits. The stochastic multi-armed bandit problem was proposed and studied by Thompson in [50]. Thompson also proposed a heuristic algorithm known as Thompson's sampling. The first series of works to give algorithms for stochastic MAB with provable guarantees was initiated by [39, 4]. While [39] proposed the algorithm `UCB` to achieve the best asymptotic regret guarantees, [4] further refined it to the algorithm `UCB1` to achieve similar regret guarantees for finite time bound. Subsequent work supplied algorithms for stochastic MAB problems in which the set of arms can be infinite and the payoff function is linear, concave, or Lipschitz-continuous; see a recent survey [16] for more background. Confidence bound techniques have been an integral part of this line of work, and they remain integral to ours.

Special cases of `BwK` and related models. Stochastic MAB problems constitute a very special case of bandits with knapsacks, in which there is only one type of resource and it is consumed deterministically at rate 1. Several papers have considered the natural generalization in which there is a single resource, with deterministic consumption, but different arms consume the resource at different rates. Guha and Munagala [30] gave a constant-factor approximation algorithm for the Bayesian case of this problem, which was later generalized by Gupta et al. [31] to settings in which the arms' reward processes need not be martingales. Tran-Thanh et al. [51, 52, 53] presented prior-free algorithms for this problem; the best such algorithm achieves a regret guarantee qualitatively similar to that of the `UCB1` algorithm.

As discussed above, several recent papers studied dynamic pricing with limited supply [13, 7, 14] and dynamic procurement on a budget [9, 47] which, in hindsight, can be cast as special cases of `BwK` featuring a two-dimensional re-

source constraint. We significantly improve over their results. Another recent paper [18] considers a special case of adjusting a repeated auction (albeit without inventory constraints).

Online primal dual. While BwK is primarily an online learning problem, it also has elements of a stochastic packing problem. The literature on prior-free algorithms for stochastic packing has flourished in recent years, starting with prior-free algorithms for the stochastic AdWords problem [20], and continuing with a series of papers extending these results from AdWords to more general stochastic packing integer programs while also achieving stronger performance guarantees [2, 21, 26, 43]. A running theme of these papers (and also of the primal-dual algorithm in this thesis) is the idea of estimating of an optimal dual vector from samples, then using this dual to guide subsequent primal decisions. Particularly relevant to our work is the algorithm of [21], in which the dual vector is adjusted using multiplicative updates, as we do in our algorithm. However, unlike the BwK problem, the stochastic packing problems considered in prior work are not learning problems: they are full information problems in which the costs and rewards of decisions in the past and present are fully known. (The only uncertainty is about the future.) As such, designing algorithms for BwK requires a substantial departure from past work on stochastic packing. Our primal-dual algorithm depends upon a hybrid of confidence-bound techniques from online learning and primal-dual techniques from the literature on solving packing LPs; combining them requires entirely new techniques for bounding the magnitude of the error terms that arise in the analysis. Moreover, our Mixture_Elimintation algorithm manages to achieve strong regret guarantees without even computing a dual solution.

Contextual bandits. Contextual bandits with arbitrary policy sets were introduced and studied by Langford and Zhang [40]. The model is a reinterpretation of the Experts setting studied by [5] along with the use of a new oracle to propose a new algorithm with improved running time. We build upon a follow up work [23] and give an algorithm for Resourceful Contextual bandits (albeit with running time which is not necessarily polynomial).

Secretary problems. One form of limitation for an algorithm is due to the online arrival of the agents. The most general model to capture this is the adversarial model, where an adversary chooses the arrival sequence of the agents and their cost in a manner that yields the worst possible outcome for the mechanism. Although in principle we would like mechanisms to be robust to such input, such a strong assumption leaves little hope for making formal performance guarantees. A popular alternative is the *secretary model* which assumes the values are chosen adversarially, though the arrival order is chosen uniformly at random from the set of all possible permutations of the agents. This is the model considered by [24] for choosing the element with the highest value from an online sequence, and has been widely used since then in various other problem domains (see e.g. [8]). Another slightly stronger assumption is that each agent is independently drawn from some unknown common distribution (see e.g. [38, 6, 13]).

Concurrent and independent work. Agrawal and Devanur [1] study a model for contextual bandits with resource constraints that is incomparable with ours. The dependence of expected rewards and resource consumptions on contexts and arms is more restrictive, namely linear. Whereas the model for resource constraints is more general, e.g., allowing diminishing returns. They also generalize our model of Bandits with Knapsacks to included concave rewards and

convex constraints.

1.5 Our contributions

1.5.1 Bandits with Knapsacks

In analyzing MAB algorithms one typically expresses the regret as a function of the time horizon, T. The regret guarantee is considered nontrivial if this function grows sublinearly as $T \to \infty$. In the BwK problem a regret guarantee of the form $o(T)$ may be unacceptably weak because supply limits prevent the optimal policy from achieving a reward close to T. An illustrative example is the dynamic pricing problem with supply $k \ll T$: the seller can only sell k items, each at a price of at most 1, so bounding the regret by any number greater than k is worthless.

We instead seek regret bounds that are sublinear in OPT, the expected reward of the optimal policy, or (at least) sublinear in M_{LP}, the maximal possible value of OPT given the budget constraints. To achieve sublinear regret, the algorithm must be able to explore each arm a significant number of times without exhausting its resource budget. Accordingly we also assume, for some $B \geq 1$, that the amount of any resource consumed in a single round is guaranteed to be no more than $1/B$ fraction of that resource's budget, and we parameterize our regret bound by B. We present an algorithm (called PD – BwK) whose regret is sublinear in OPT as both OPT and B tend to infinity. More precisely, denoting the

number of arms by m, our algorithm's regret is

$$\widetilde{O}\left(\sqrt{m\,\texttt{OPT}} + \texttt{OPT}\sqrt{m/B}\right). \tag{1.1}$$

In particular, if all budget constraints (including the time horizon) are scaled up by the factor of $\alpha > 1$, regret scales as $\sqrt{\alpha}$. The algorithm is computationally efficient, in a strong sense: the per-round running time is polynomial in the number of arms and the number of resources. We also present another algorithm (called `Mixture_Elimintation`) whose regret bound is qualitatively similar; instead of depending on `OPT` it depends on M_{LP}. The two algorithms use quite different techniques.

Algorithm `PD-BwK` is a primal-dual algorithm based on the multiplicative weights update method. It maintains a vector of "resource costs" that is adjusted using multiplicative updates. In every period it estimates each arm's expected reward and expected resource consumption, using upper confidence bounds for the former and lower confidence bounds for the latter; then it plays the most "cost-effective" arm, namely the one with the highest ratio of estimated resource consumption to estimated resource cost, using the current cost vector. Although confidence bounds and multiplicative updates are the bread and butter of online learning theory, we consider this way of combining the two techniques to be quite novel. In particular, previous multiplicative-update algorithms in online learning theory — such as the `Exp3` algorithm for MAB [5] or the weighted majority [41] and `Hedge` [28] algorithms for learning from expert advice — applied multiplicative updates to the probabilities of choosing different arms (or experts). Our application of multiplicative updates to the dual variables of the LP relaxation of `BwK` is conceptually quite a different usage of this technique.

Theorem 1.5.1. *For any instance of the* BwK *problem, the regret of Algorithm* PD − BwK *satisfies*

$$\texttt{OPT} - \texttt{REW} \leq O\left(\sqrt{\log(dmT)}\right)\left(\sqrt{m\,\texttt{OPT}} + \texttt{OPT}\,\sqrt{\frac{m}{B}} + m\sqrt{\log(dmT)}\right). \tag{1.2}$$

Algorithm Mixture_Elimintation explicitly optimizes over mixtures of arms, based on a very simple idea: balanced exploration inside confidence bounds. The design principle underlying confidence-bound based algorithms for stochastic MAB (including the famous UCB1 algorithm [4] and our algorithm PD − BwK) is generally, "Exploit as much as possible, but use confidence bounds that are wide enough to encourage some exploration." Our algorithm's design principle, in contrast, could be summarized as, "Explore as much as possible, but use confidence bounds that are narrow enough to eliminate obviously suboptimal alternatives." In fact, Mixture_Elimintation balances its exploration across arms, so that (essentially) *each arm* is explored as much as possible. More precisely, for each arm, there are designated rounds when, once the obviously suboptimal mixtures of arms are eliminated, the algorithm picks a mixture that approximately maximizes the probability of choosing this arm. Intriguingly, the algorithm matches the logarithmic regret bound of UCB1 for stochastic MAB (up to constant factors) and achieves qualitatively similar bounds to PD − BwK for BwK, despite being based on a design principle that is the polar opposite of those algorithms. We believe that the "balanced exploration" principle underlying Mixture_Elimintation is a novel and important conceptual contribution to the theory of MAB algorithms that is likely to find other applications.

Theorem 1.5.2 (BwK: Mixture_Elimintation)**.** *Consider an instance of* BwK *with d resources, $m = |X|$ arms, and the smallest budget $B = \min_i B_i$. Let $M_{\texttt{LP}}$ be the maximum of $\texttt{OPT}_{\texttt{LP}}$, for given time horizon and budgets, over all problem instances in a given BwK*

domain. Algorithm `Mixture_Elimintation` *achieves total expected regret*

$$\texttt{OPT} - \texttt{REW} \leq O(\log(dmT)\log(T/m))\left(\sqrt{dm\,M_{\text{LP}}} + M_{\text{LP}}\left(\sqrt{\frac{dm}{B}} + \frac{dm}{B}\right) + dm\right). \quad (1.3)$$

Further, we provide a matching lower bound: we prove that regret (1.1) is optimal up to polylog factors. Specifically, we show that any algorithm for `BwK` must incur regret

$$\Omega\left(\min\left(\texttt{OPT},\ \texttt{OPT}\sqrt{m/B} + \sqrt{m\,\texttt{OPT}}\right)\right), \quad (1.4)$$

in the worst-case over all instances of `BwK` with given $(m, B, \texttt{OPT})$. We derive this lower bound using a simple example in which all arms have reward 1 and 0-1 consumption of a single resource, and one arm has slightly smaller expected resource consumption than the rest. To analyze this example, we apply the KL-divergence technique from the MAB lower bound in [5]. Some technical difficulties arise (compared to the derivation in [5]) because the arms are different in terms of the expected consumption rather than expected reward, and because we need to match the desired value for `OPT`.

Theorem 1.5.3. *Informally, any algorithm for* `BwK` *must incur regret*

$$\Omega\left(\min\left(\texttt{OPT},\ \texttt{OPT}\sqrt{\frac{m}{B}} + \sqrt{m\,\texttt{OPT}}\right)\right), \quad (1.5)$$

where $m = |X|$ is the number of arms and $B = \min_i B_i$ is the smallest budget.

More precisely, fix any $m \geq 2$, $d \geq 1$, $\texttt{OPT} \geq m$, and $(B_1, \ldots, B_d) \in [2, \infty)$. Let $\mathcal{F}$ be the family of all `BwK` *problem instances with m arms, d resources, budgets $(B_1, \ldots, B_d)$ and optimal reward* `OPT`*. Then any algorithm for* `BwK` *must incur regret (3.29) in the worst case over $\mathcal{F}$.*

Applications and special cases. Due to its generality, the `BwK` problem admits applications in diverse domains such as dynamic pricing, dynamic procure-

ment, ad allocation, repeated auctions, network routing, and scheduling. These applications are discussed in Section 3.7; below we provide some highlights.

The BwK setting subsumes dynamic pricing with limited supply, as studied in [15, 35, 13, 7, 14].[2] Specializing our regret bounds to this setting, we obtain the optimal regret $\tilde{O}(k^{2/3})$ with respect to the optimal policy, where k is the number of items. The prior work [7] achieved the same regret bound with respect to the best fixed price, which is a much weaker benchmark, and proved a matching lower bound. Further, our setting allows to incorporate a number of generalizations such as selling multiple products (with a limited supply of each), volume pricing, pricing bundles of goods, and profiling of buyers according to their types. In particular, we improve over the result in [14] for multiple products.

A "dual" problem to dynamic pricing is *dynamic procurement* [9, 47], where the algorithm is "dynamically buying" rather than "dynamically selling". (The budget constraint now applies to the amount spent rather than the quantity of items sold, which is why the problems are not merely identical up to sign reversal.) This problem is particularly relevant to the emerging domain of *crowdsourcing*: the items "bought" then correspond to microtasks ordered on a crowdsourcing platform such as Amazon Turk. We obtain significant improvements for the basic version of dynamic procurement studied in [9, 47]. In particular, [9] achieves a constant-factor approximation to the optimum with a prohibitively large constant (at least in the tens of thousands), whereas our approximation factor $1 + \tilde{O}((T/\texttt{OPT})B^{-1/4})$ is typically much smaller. The regret bound in [47] is against the best-fixed-price benchmark, which may be much smaller than OPT

[2]The Bayesian version of this problem has a rich literature in Operations Research, see [13] for an overview. The earlier papers [15, 35] on the regret-minimizing version focus on the special case of unlimited supply.

(see Section 3.9).[3] Furthermore, the generality of BwK allows to incorporate generalizations such as multiple types of items/microtasks, and the presence of other, competing algorithms.

Further, BwK applies to the problem of dynamic ad allocation, in the context of pay-per-click advertising with unknown click probabilities. It allows to extend a standard (albeit idealized) model of ad allocation to incorporate advertisers' budgets. In fact, BwK allows advertisers to specify multiple budget constraints on possibly overlapping subsets of ads.

Discussion. If the BwK instance includes multiple, different budget constraints, our regret bounds are with respect to the *smallest* of these constraints. This, however, is inevitable for the worst-case regret bounds.

Algorithm Mixture_Elimintation is *domain-aware*, in the sense that it explicitly optimizes over all latent distributions that are feasible for a given BwK domain (such as dynamic pricing with limited supply). We do not provide a computationally efficient implementation for this optimization. In fact, it is not clear what model of computation would be appropriate to characterize access to the domain knowledge. Efficient implementation of Mixture_Elimintation may be possible for some specific BwK domains, although we do not pursue that direction in this thesis. (Recall that PD – BwK, on the other hand, is very computationally efficient.)

It is worth noting that our regret bounds for PD – BwK are stronger than that of Mixture_Elimintation. But, we are able to generalize the second algorithm

[3]The paper [47] is simultaneous with respect to our paper. One cannot directly compare our regret bound and theirs (benchmarks aside), because [47] does not derive a worst-case regret bound.

to more general settings in the next model we study.

1.5.2 Resourceful contextual bandits

We note that the model of RCB is a generalization of BwK. Hence, any algorithm we develop form RCB should be also applicable for BwK. We generalize the algorithm Mixture_Elimintation for BwK to algorithm Contexutal_Mixture_Elimination for RCB.

The algorithm at a very high level can be described as follows. At each step we maintain a estimates along with confidence bounds similar to other algorithms based on confidence bounds. We avoid/eliminate obviously sub-optimal strategies subject to confidence bounds. Subject to the previous constraint we pick strategies which explores as much as possible. We make the algorithm Contexutal_Mixture_Elimination formal in chapter 4.

Theorem 1.5.4. *For all* RCB *problems with K actions, d resources, time horizon T, and for all policy sets Π. Algorithm* Contexutal_Mixture_Elimination *achieves regret*

$$O\left(1 + \tfrac{1}{B}\,\mathtt{OPT}(\Pi)\right)\sqrt{dKT\ \log\left(dKT\,|\Pi|\right)},$$

where $B = \min_i B_i$ is the smallest resource constraint.

There are several interesting aspects to the regret bound of theorem 1.5.4 which we note here.

- The regret bounds is not dependent on the size of the context space. This allows us to consider very high dimensional context spaces. Such a property is very useful because it can model problems where we need to track cookie information of users in a browser with several variables.

- The regret bound is only logarithmically dependent on the number of policies. One can think of these policies as the setting of parameters in a machine learning model and we can throw in a lot of these models because of the logarithmic dependence.
- Unlike `Mixture_Elimintation` the regret bound for `Contexutal_Mixture_Elimination` depends on T. This is not very desirable because when $B < \sqrt{kT}$ we get no non-trivial regret guarantees. Unfortunately, we show in theorem 1.5.5 that such a dependence is in fact necessary.

Theorem 1.5.5. *No algorithm for* `RCB` *can achieve regret* $o(\texttt{OPT}(\Pi))$ *for all problem instances such that* $\texttt{OPT}(\Pi) \leq B \leq \sqrt{KT}/2$ *(using the notation from Theorem 1.5.4).*

1.5.3 Dynamic procurement

While our models for `BwK` and `RCB` are very general and capture many settings, they have two drawbacks. One is the fact that they consider the rewards which add up over rounds and another is that they only achieve an additive regret guarantee. For the special case of dynamic procurement we are able to overcome these and achieve the desired results.

Our first result is for the special case of symmetric submodular procurement market. For this purpose we introduce a novel random walk based algorithm which achieves a constant factor approximation.

Theorem 1.5.6. *For symmetric submodular procurement market where the agents arrive from an unknown distribution the mechanism* `RAND_WALK` *outputs a solution with an expected utility which is a constant factor approximation to the expected optimal solutions which knows the true values of agents in advance.*

Our second result is to get a more general result when the buyer has a general submodular function. We propose an algorithm which uses binning and guessing to get a $O(\log n)$ approximation for this problem.

Theorem 1.5.7. *For submodular procurement market where the agents arrive in a random order the mechanism* `GUESS_POST` *outputs a solution with an expected utility which is a* $O(\log n)$ *factor approximation to the expected optimal solutions which knows the true values of agents in advance.*

1.6 Outline of this thesis

The thesis is divided into three main chapters. In chapter 2 we give the necessary preliminaries. In chapter 3 we introduce the model of Bandits with Knapsacks and give tight upper and lower bounds. In chapter 4 we introduce resource constrained contextual bandits. In chapter 5 we study the special case of dynamic procurement and give multiplicative approximations to this problem.

1.7 Bibliogrpahic Notes

The results of this thesis appeared in the following papers: Badanidiyuru, Kleinberg and Slivkins [10](FOCS 2013), Badanidiyuru, Langford and Slivkins [12](COLT 2014) and Badanidiyuru, Kleinberg and Singer [9](EC 2012). The results in chapter 3 are from [10], in chapter 4 are from [12] and in chapter 5 are from [9].

CHAPTER 2

BACKGROUND MATERIAL

In this chapter we will cover some of the background material which will be useful for us.

2.1 Analysis of the Hedge Algorithm

In this section we analyze the Hedge algorithm, also known as the multiplicative weights algorithm. It is an online algorithm for maintaining a d-dimensional probability vector y while observing a sequence of d-dimensional payoff vectors $\pi_1, \ldots, \pi_\tau$. The algorithm is initialized with a parameter $\epsilon \in (0, 1)$.

Algorithm 1: The algorithm Hedge(ϵ)

1: $v_1 = \mathbf{1}$

2: **for** $t = 1, 2, \ldots, \tau$ **do**

3: $\quad y_t = v_t/(\mathbf{1}^\intercal v_t)$.

4: $\quad v_{t+1} = \mathrm{Diag}\{(1+\epsilon)^{\pi_{ti}}\}v_t$.

The performance guarantee of the algorithm is expressed by the following proposition.

Proposition 2.1.1. *For any* $0 < \epsilon < 1$ *and any sequence of payoff vectors* $\pi_1, \ldots, \pi_\tau \in [0, 1]^d$*, we have*

$$\forall y \in \mathbf{\Delta}[d] \quad \sum_{t=1}^{\tau} y_t^\intercal \pi_t \geq (1-\epsilon) \sum_{t=1}^{\tau} y^\intercal \pi_t - \frac{\ln d}{\epsilon}.$$

Proof. The analysis uses the potential function $\Phi_t = \mathbf{1}^\intercal v_t$. We have

$$\begin{aligned}
\Phi_{t+1} &= \mathbf{1}^\intercal \mathrm{Diag}\{(1+\epsilon)^{\pi_{ti}}\} v_t \\
&= \sum_{i=1}^{d} (1+\epsilon)^{\pi_{ti}} v_{ti} \\
&\leq \sum_{i=1}^{d} (1+\epsilon\pi_{ti}) v_{ti} \\
&= \Phi_t \left(1 + \epsilon y_t^\intercal \pi_t\right) \\
\ln(\Phi_{t+1}) &\leq \ln(\Phi_t) + \ln(1+\epsilon y_t^\intercal \pi_t) \leq \ln(\Phi_t) + \epsilon y_t^\intercal \pi_t.
\end{aligned}$$

On the third line, we have used the inequality $(1+\epsilon)^x \leq 1 + \epsilon x$ which is valid for $0 \leq x \leq 1$. Now, summing over $t = 1, \ldots, \tau$ we obtain

$$\sum_{t=1}^{\tau} y_t^\intercal \pi_t \geq \tfrac{1}{\epsilon} (\ln \Phi_{\tau+1} - \ln \Phi_1) = \tfrac{1}{\epsilon} \ln \Phi_{\tau+1} - \frac{\ln d}{\epsilon}.$$

The maximum of $y^\intercal \left(\sum_{t=1}^{\tau} \pi_t\right)$ over $y \in \mathbf{\Delta}[d]$ must be attained at one of the extreme points of $\mathbf{\Delta}[d]$, which are simply the standard basis vectors of $\mathbb{R}^d$. Say that the maximum is attained at $\mathbf{e}_i$. Then we have

$$\begin{aligned}
\Phi_{\tau+1} = \mathbf{1}_\intercal v_{\tau+1} \geq v_{\tau+1,i} &= (1+\epsilon)^{\pi_{1i}+\cdots+\pi_{\tau i}} \\
\ln \Phi_{\tau+1} &\geq \ln(1+\epsilon) \sum_{t=1}^{\tau} \pi_{ti} \\
\sum_{t=1}^{\tau} y_t^\intercal \pi_t &\geq \frac{\ln(1+\epsilon)}{\epsilon} \sum_{t=1}^{\tau} \pi_{ti} - \frac{\ln d}{\epsilon} \\
&\geq (1-\epsilon) \sum_{t=1}^{\tau} y^\intercal \pi_t - \frac{\ln d}{\epsilon}.
\end{aligned}$$

The last line follows from two observations. First, our choice of i ensures that $\sum_{t=1}^{\tau} \pi_{ti} \geq \sum_{t=1}^{\tau} y^\intercal \pi_t$ for every $y \in \mathbf{\Delta}[d]$. Second, the inequality $\ln(1+\epsilon) > \epsilon - \epsilon^2$ holds for every $\epsilon > 0$. In fact,

$$\begin{aligned}
-\ln(1+\epsilon) = \ln\left(\frac{1}{1+\epsilon}\right) = \ln\left(1 - \frac{\epsilon}{1+\epsilon}\right) &< -\frac{\epsilon}{1+\epsilon} \\
\ln(1+\epsilon) > \frac{\epsilon}{1+\epsilon} &> \frac{\epsilon(1-\epsilon^2)}{1+\epsilon} = \epsilon - \epsilon^2.
\end{aligned}$$

□

2.2 High probability events

All of our models are based on stochastic assumptions. Under these assumptions any repeated event has good concentration properties and we can take advantage of this. We state here the specific lemmas which we will using in our results.

We will first state three lemma in increasing order or generality to deal with independent random variables. These are standard inequalities and can be found in [22]. Then we will state a lemma to handle dependent random variables.

Lemma 2.2.1 (Markov's inequality)**.** *Let X be a random varibale which takes values in $[0, \infty]$ and has expectation μ. Then, for any $w > 0$*

$$\Pr\left[X \geq w\,\mu\right] \leq \frac{1}{w}$$

Lemma 2.2.2 (Chebyshev's inequality)**.** *Let X be a random varibale which takes values in $[0, \infty]$ and has expectation μ and variance σ^2. Then, for any $w > 0$*

$$\Pr\left[|X - \mu| \geq w\,\sigma\right] \leq \frac{1}{w^2}$$

Lemma 2.2.3 (Chernoff Bound)**.** *Let $X_1, \ldots, X_\ell$ be independent random variables where X_i take values in $[0, w_i]$ and let $\mu = \mathbb{E}[\sum_{i=1}^{\ell} X_i]$. Then, for any $\delta > 0$ we have that:*

$$Pr\Big[\sum_{i=1}^{\ell} X_i \geq (1+\delta)\mu\Big] \leq \quad (\tfrac{e^{\delta}}{(1+\delta)^{(1+\delta)}})^{\frac{\mu}{max_i w_i}} \tag{2.1}$$

$$Pr\Big[\sum_{i=1}^{\ell} X_i \leq (1-\delta)\mu\Big] \leq \qquad e^{\frac{-\delta^2\mu}{2\cdot max_i w_i}} \tag{2.2}$$

The next concentration bound is to handle dependent events. This is Bernstein's inequality for martingales [27], via the following formulation from [17]:

Lemma 2.2.4. *Let $\mathcal{G}_1 \subseteq \ldots \subseteq \mathcal{G}_n$ be a filtration, and $X_1, \ldots, X_n$ be real random variables such that X_t is $\mathcal{G}_t$-measurable, $\mathbb{E}(X_t|\mathcal{G}_{t-1}) = 0$ and $|X_t| \leq b$ for some $b > 0$. Let $V_n = \sum_{t=1}^{n} \mathbb{E}(X_t^2|\mathcal{G}_{t-1})$. Then with probability at least $1-\delta$ it holds that*

$$\textstyle\sum_{t=1}^{n} X_t \leq \sqrt{4V_n \log(n\delta^{-1}) + 5b^2 \log^2(n\delta^{-1})}.$$

2.3 Kullback-Leibler Divergence

The proof of Lemma 3.6.5 relies on the concept of KL-divergence. Let us provide some background to make on KL-divergence which will be used in chapter 3 to prove lowerbound. We use a somewhat non-standard notation that is tailored to the needs of our analysis.

The *KL-divergence* (a.k.a. *relative entropy*) is defined as follows. Consider two distributions μ, ν on the same finite universe Ω. Assume $\mu \ll \nu$ (in words, μ is *absolutely continuous* with respect to ν), meaning that $\nu(w) = 0 \Rightarrow \mu(w) = 0$ for all $w \in \Omega$. Then KL-divergence of μ given ν is

$$\mathtt{KL}(\mu \,\|\, \nu) \triangleq \mathop{\mathbb{E}}_{w\sim(\Omega,\mu)} \log\left(\frac{\mu(w)}{\nu(w)}\right) = \sum_{w\in\Omega} \log\left(\frac{\mu(w)}{\nu(w)}\right)\mu(w).$$

In this formula we adopt a convention that $\frac{0}{0} = 1$. We will use the fact that

$$\mathtt{KL}(\mu \,\|\, \nu) \geq \tfrac{1}{2} \,\|\mu - \nu\|_1^2. \tag{2.3}$$

Henceforth, let μ, ν be distributions on the universe Ω^∞, where Ω is a finite set. For $\vec{w} = (w_1, w_2, \ldots) \in \Omega^\infty$ and $t \in \mathbb{N}$, let us use the notation $\vec{w}_t = (w_1, \ldots, w_t) \in \Omega^t$. Let μ_t be a restriction of μ to Ω^t: that is, a distribution on Ω^t given by

$$\mu_t(\vec{w}_t) \triangleq \mu\left(\{\vec{u} \in \Omega^\infty : \ \vec{u}_t = \vec{w}_t\}\right).$$

The *next-round conditional distribution* of μ given $\vec{w}_t$, $t < T$ is defined by

$$\mu\left(w_{t+1} \,|\, \vec{w}_t\right) \triangleq \frac{\mu_{t+1}(\vec{w}_{t+1})}{\mu_t(\vec{w}_t)}.$$

Note that $\mu(\cdot\,|\vec{w}_t)$ is a distribution on Ω for every fixed $\vec{w}_t$.

The *conditional KL-divergence* at round $t+1$ is defined as

$$\mathtt{KL}_{t+1}(\mu \,\|\, \nu) \triangleq \mathop{\mathbb{E}}_{\vec{w}_t \sim (\Omega^t,\, \mu_t)} \mathtt{KL}(\mu(\cdot\,|\vec{w}_t) \,\|\, \nu(\cdot\,|\vec{w}_t)).$$

In words, this is the KL-divergence between the next-round conditional distributions $\mu(\cdot\,|\vec{w}_t)$ and $\nu(\cdot\,|\vec{w}_t)$, in expectation over the random choice of $\vec{w}_t$ according to distribution μ_t.

We will use the following fact, known as the *chain rule* for KL-divergence:

$$\mathtt{KL}(\mu_T \,\|\, \nu_T) = \sum_{t=1}^{T} \mathtt{KL}_t(\mu \,\|\, \nu), \quad \text{for each } T \in \mathbb{N}. \tag{2.4}$$

Here for notational convenience we define $\mathtt{KL}_1(\mu \,\|\, \nu) \triangleq \mathtt{KL}(\mu_1 \,\|\, \nu_1)$.

2.4 Useful lemmas

We will prove useful facts which will be used in the later sections.

Fact 2.4.1. *Let S_t be the sum of t i.i.d. 0-1 variables with expectation q. Let τ be the first time this sum reaches a given number $B \in \mathbb{N}$. Then $\mathbb{E}[\tau] = B/q$. Moreover, for each $T > \mathbb{E}[\tau]$ it holds that*

$$\sum_{t>T} \Pr[\tau \geq t] \leq \mathbb{E}[\tau]^2/T. \tag{2.5}$$

Proof. $\mathbb{E}[\tau] = B/q$ follows from the martingale argument presented in the proof of Claim 3.6.3. Formally, take $q = p - \epsilon$ and $N_\tau = \tau$.

Assume $T > \mathbb{E}[\tau]$. The proof of Equation (2.5) uses two properties, one being that a geometric random variable is memoryless and other being Markov's inequality. Let us first bound the random variable $\tau - T$ conditional on the event that $\tau > T$.

$$\begin{aligned}
\mathbb{E}[\tau - T|\tau > T] &= \textstyle\sum_{t=1}^{B} \Pr[S_T = t] \ \mathbb{E}[\tau - T|\tau > T, S_T = t] \\
&= \textstyle\sum_{t=1}^{B} \Pr[S_T = t] \ \mathbb{E}[\tau - T|S_T = t] \\
&\leq \textstyle\sum_{t=1}^{B} \Pr[S_T = t] \ \mathbb{E}[\tau - T|S_T = 0] \\
&\leq \mathbb{E}[\tau - T|S_T = 0] = \mathbb{E}[\tau].
\end{aligned}$$

By Markov's inequality we have $\Pr[\tau \geq T] \leq \frac{\mathbb{E}[\tau]}{T}$. Combining the two inequalities, we have

$$\begin{aligned}
\textstyle\sum_{t>T} \Pr[\tau \geq t] &= \textstyle\sum_{t>T} \Pr[\tau \geq T] \ \ \Pr[\tau \geq t|\tau > T] \\
&= \Pr[\tau \geq T] \ \ \mathbb{E}[\tau - T|\tau > T] \\
&\leq \mathbb{E}[\tau]^2/T.
\end{aligned}$$

□

Fact 2.4.2. *Assume $\frac{\epsilon}{p} \leq \frac{1}{2}$ and $p \leq \frac{1}{2}$. Then*

$$p \log\left(\frac{p}{p-\epsilon}\right) + (1-p) \log\left(\frac{1-p}{1-p+\epsilon}\right) \leq \frac{2\epsilon^2}{p}.$$

Proof. To prove the inequality we use the following standard inequalities:

$$\log(1+x) \geq x - x^2/2 \qquad \forall x \in [0,1]$$

$$\log(1-x) \geq -x - x^2 \qquad \forall x \in [0, \tfrac{1}{2}].$$

It follows that:

$$\begin{aligned} p \log\left(\frac{p}{p-\epsilon}\right) + (1-p)\log\left(\frac{1-p}{1-p+\epsilon}\right) &= -p\log\left(1-\frac{\epsilon}{p}\right) - (1-p)\log\left(1+\frac{\epsilon}{1-p}\right) \\ &\leq p\left(\frac{\epsilon}{p} + \frac{\epsilon^2}{p^2}\right) + (1-p)\left(-\frac{\epsilon}{1-p} + \frac{\epsilon^2}{(1-p)^2}\right) \\ &= \frac{\epsilon^2}{p} + \frac{\epsilon^2}{1-p} \leq \frac{2\epsilon^2}{p} \end{aligned}$$

□

CHAPTER 3

BANDITS WITH KNAPSACKS

In this chapter we consider the BwK problem. We propose a pair of algorithms for solving the problem: a "balanced exploration" algorithm (Mixture_Elimintation) that prioritizes information acquisition, exploring each arm as frequently as possible given current confidence intervals, and a primal-dual algorithm (PD – BwK) that chooses arms to greedily maximize the estimated reward per unit of resource cost. We further show that the performance guarantees of both the algorithms are near optimal. While PD – BwK has a better regret guarantee with respect to certain parameters and has a polynomial running time, we will show in the chapter 4 that Mixture_Elimintation is generalizable to the contextual setting.

In section 3.1 we introduce the tools needed for our result. Then in section 3.2 we will give the first algorithm PD – BwK and then present its analysis in section 3.3. In section 3.4 we will give the second algorithm Mixture_Elimintation and present its analysis in section 3.5. We complement the algorithms by showing a nearly tight lowerbound in section 3.6. We end the section with showing how the applications fit formally in the model in section 3.7.

3.1 Tools

In the classic stochastic multi-armed bandit problem without resources the benchmark is the maximum total expected reward of any fixed arm. But as we will show examples in section 3.9 this is not the case with BwK. So in order to get a handle on the OPT we will need tools that we introduce in this section.

3.1.1 LP-relaxation

The expected reward of the optimal policy, given foreknowledge of the distribution of outcome vectors, is typically difficult to characterize exactly. In fact, even for a time-invariant policy, it is difficult to give an exact expression for the expected reward due to the dependence of the reward on the random stopping time, τ, when the resource budget is exhausted. To approximate these quantities, we consider the fractional relaxation of BwK in which the stopping time (i.e., the total number of rounds) can be fractional, and the reward and resource consumption per unit time are deterministically equal to the corresponding expected values in the original instance of BwK.

The following LP (shown here along with its dual) constitutes our fractional relaxation of the optimal policy. We denote by $\texttt{OPT}_{\texttt{LP}}$ the value of the linear

$$\begin{array}{lll} \max & r^\intercal \xi & \\ \text{s.t.} & C\xi \preceq B\mathbf{1} & \\ & \xi \geq 0 & \end{array} \qquad (\text{P}) \qquad\qquad \begin{array}{lll} \min & B\mathbf{1}^\intercal \eta & \\ \text{s.t.} & C^\intercal \eta \succeq r & \\ & \eta \geq 0 & \end{array} \qquad (\text{D})$$

program (P).

Lemma 3.1.1. $\texttt{OPT}_{\texttt{LP}}$ *is an upper bound on the value of the optimal dynamic policy.*

Proof. One way to prove the lemma is to define ξ_j to be the expected number of times arm a_j is played by the optimal dynamic policy, and argue that ξ is primal-feasible and that $r^\intercal \xi$ is the expected reward of the optimal policy. We instead present a simple proof using the dual LP (D), since it introduces ideas that motivate the design of our primal-dual algorithm.

Let η^* denote an optimal solution to (D). By strong duality, $B\mathbf{1}^\intercal\eta^* = \texttt{OPT}_{\texttt{LP}}$. Interpret η_i^* as a unit cost for resource i. Dual feasibility implies that for each arm a_j, the expected cost of resources consumed when a_j is pulled exceeds the expected reward produced. Thus, if we let Z_t denote the sum of rewards gained in rounds $1, \ldots, t$ plus the cost of the remaining resource endowment after round t, then the stochastic process $Z_0, Z_1, \ldots, Z_T$ is a supermartingale. Note that $Z_0 = B\mathbf{1}^\intercal\eta^*$ is the LP optimum, and $Z_{\tau-1}$ equals the algorithm's total payoff, plus the cost of the remaining (non-negative) resource supply at the start of round τ. By Doob's optional stopping theorem, $Z_0 \geq \mathbb{E}[Z_{\tau-1}]$ and the lemma is proved. □

In a similar way, the expected total reward of time-invariant policy $\mathcal{D}$ is bounded above by the solution to the following linear program in which t is the only LP variable:

$$\begin{array}{lrcll} \text{Maximise} & t\, r(\mathcal{D},\mu) & & & \text{in } t \in \mathbb{R} \\ \text{subject to} & t\, c_i(\mathcal{D},\mu) & \leq & B & \text{for each resource } i \\ & t & \geq & 0. & \end{array} \tag{3.1}$$

The solution to (3.1), which we call the *LP-value*, is

$$\texttt{LP}(\mathcal{D},\mu) = r(\mathcal{D},\mu)\ \min_i\left(\frac{B}{c_i(\mathcal{D},\mu)}\right). \tag{3.2}$$

Observe that t is feasible for $\texttt{LP}(\mathcal{D},\mu)$ if and only if $\xi = t\mathcal{D}$ is feasible for (P), and therefore

$$\texttt{OPT}_{\texttt{LP}} = \sup_{\mathcal{D}} \texttt{LP}(\mathcal{D},\mu).$$

A distribution $D^* \in \operatorname{argmax}_{\mathcal{D}} \texttt{LP}(\mathcal{D},\mu)$ is called *LP-optimal* for latent structure μ. Any optimal solution ξ to (P) corresponds to an LP-optimal distribution $D^* = \xi/\|\xi\|_1$.

Claim 3.1.2. *For any latent structure μ, there exists a distribution $\mathcal{D}$ over arms which is LP-optimal for μ and moreover satisfies the following three properties:*

(a) $c_i(\mathcal{D}, \mu) \leq B/T$ *for each resource* i.

(b) $\mathcal{D}$ *has a support of size at most* $d + 1$.

(c) *If* $\mathcal{D}$ *has a support of size at least* 2 *then for some resource* i *we have* $c_i(\mathcal{D}, \mu) = B/T$.

(Such distribution $\mathcal{D}$ *will be called* ***LP-perfect*** *for* μ*.)*

Proof. Fix the latent structure μ. It is a well-known fact that for any linear program there exists an optimal solution whose support has size that is exactly equal to the number of constraints that are tight for this solution. Take any such optimal solution ξ for linear program (P), and take the corresponding LP-optimal distribution $\mathcal{D} = \xi/\|\xi\|_1$. Since there are $d + 1$ constraints in (P), distribution $\mathcal{D}$ has support of size at most $d + 1$. If it satisfies property (a), we are done. Note that such $\mathcal{D}$ also satisfies property (c).

Suppose property (a) does not hold for $\mathcal{D}$. Then there exists a resource i such that $c_i(\mathcal{D}, \mu) > B/T$. It follows that $\sum_i \xi_i < T$. Therefore, at most d constraints in (P) are tight for ξ, which implies that the support of $\mathcal{D}$ has size at most d.

Now, let us modify $\mathcal{D}$ to obtain another LP-optimal distribution $\mathcal{D}'$ which satisfies both (a) and (b-c). Namely, let $\mathcal{D}'(a) = \alpha\, \mathcal{D}(a)$ for each non-null arm a, for some $\alpha \in (0, 1)$ that is sufficiently low to ensure property (a) (with equality for some resource i), and place the remaining probability in $\mathcal{D}'$ on the null arm. Then $\mathtt{LP}(\mathcal{D}', \mu) = \mathtt{LP}(\mathcal{D}, \mu)$, so $\mathcal{D}'$ is LP-optimal; $\mathcal{D}'$ satisfies properties (a,c) by design, and it satisfies property (b) because it adds at most one to the support of $\mathcal{D}$. □

3.1.2 High-probability events

We will use the following expression, which we call the *confidence radius*.

$$\texttt{rad}(\nu, N) = \sqrt{\frac{C_{\texttt{rad}}\,\nu}{N}} + \frac{C_{\texttt{rad}}}{N}. \tag{3.3}$$

Here $C_{\texttt{rad}} = \Theta(\log(d\,T|A|))$ is a parameter which we will fix later; we will keep it implicit in the notation. The meaning of Equation (3.3) and $C_{\texttt{rad}}$ is explained by the following tail inequality from [36, 7].[1]

Theorem 3.1.3 ([36, 7]). *Consider some distribution with values in* $[0,1]$ *and expectation* ν*. Let* $\widehat{\nu}$ *be the average of* N *independent samples from this distribution. Then*

$$\Pr\left[\ |\nu - \widehat{\nu}| \leq \texttt{rad}(\widehat{\nu}, N) \leq 3\,\texttt{rad}(\nu, N)\ \right] \geq 1 - e^{-\Omega(C_{\texttt{rad}})}, \quad \text{for each } C_{\texttt{rad}} > 0. \tag{3.4}$$

More generally, Equation (3.4) holds if $X_1, \ldots, X_N \in [0,1]$ *are random variables,* $\widehat{\nu} = \frac{1}{N}\sum_{t=1}^{N} X_t$ *is the sample average, and* $\nu = \frac{1}{N}\sum_{t=1}^{N} \mathbb{E}[X_t \mid X_1, \ldots, X_{t-1}]$.

If the expectation ν is a latent quantity, Equation (3.4) allows us to estimate ν by a high-confidence interval

$$\nu \in [\widehat{\nu} - \texttt{rad}(\widehat{\nu}, N),\ \widehat{\nu} + \texttt{rad}(\widehat{\nu}, N)], \tag{3.5}$$

whose endpoints are observable (known to the algorithm). For small ν, such estimate is much sharper than the one provided by Azuma-Hoeffding inequality.[2] Our algorithm uses several estimates of this form. For brevity, we say that $\widehat{\nu}$ is an *N-strong estimate* of ν if $\nu, \widehat{\nu}$ satisfy Equation (3.5).

It is sometimes useful to argue about *any* ν which lies in the high-confidence interval (3.5), not just the latent $\nu = \mathbb{E}[\widehat{\nu}]$. We use the following claim which is implicit in [36].

[1] Specifically, this follows from Lemma 4.9 in the full version of [36] (on arxiv.org), and Theorem 4.8 and Theorem 4.10 in the full version of [7] (on arxiv.org).

[2] Essentially, Azuma-Hoeffding inequality states that $|\nu - \widehat{\nu}| \leq O(\sqrt{C_{\texttt{rad}}/N})$, whereas by Theorem 3.1.3 for small ν it holds with high probability that $\texttt{rad}(\widehat{\nu}, N) \sim C_{\texttt{rad}}/N$.

Claim 3.1.4 ([36]). *For any* $v, \widehat{v} \in [0,1]$, *Equation (3.5) implies that* $\texttt{rad}(\widehat{v}, N) \leq 3\,\texttt{rad}(v, N)$.

3.2 Algorithm: PD − BwK

This section develops an algorithm that solves the BwK problem using a very natural and intuitive idea: greedily select arms with the greatest estimated "bang per buck," i.e. reward per unit of resource consumption. One of the main difficulties with this idea is that there is no such thing as a "unit of resource consumption": there are d different resources, and it is unclear how to trade off consumption of one resource versus another. The proof of Lemma 3.1.1 gives some insight into how to quantify this trade-off: an optimal dual solution η^* can be interpreted as a vector of unit costs for resources, such that for every arm the expected reward is less than or equal to the expected cost of resources consumed. Since our goal is to match the optimum value of the LP as closely as possible, we must minimize the shortfall between the expected reward of the arms we pull and their expected resource cost as measured by η^*. Thus, our algorithm will try to learn an optimal dual vector η^* in tandem with learning the latent structure μ.

Borrowing an idea from [3, 29, 46], we will use the multiplicatives weights update method to learn the optimal dual vector. This method raises the cost of a resource exponentially as it is consumed, which ensures that heavily demanded resources become costly, and thereby promotes balanced resource consumption. Meanwhile, we still have to ensure (as in any multi-armed bandit problem) that our algorithm explores the different arms frequently enough to gain adequately

accurate estimates of the latent structure. We do this by estimating rewards and resource consumption as optimistically as possible, i.e. using upper confidence bound (UCB) estimates for rewards and lower confidence bound (LCB) estimates for resource consumption. Although both of these techniques — multiplicative weights and confidence bounds — have both successfully applied in previous online learning algorithms, it is far from obvious that this particular hybrid of the two methods should be effective. In particular, the use of multiplicative updates on dual variables, rather than primal ones, distinguishes our algorithm from other bandit algorithms that use multiplicative weights (e.g. the `Exp3` algorithm [5]) and brings it closer in spirit to the literature on stochastic packing algorithms (especially [21]).

The pseudocode for the algorithm is presented as Algorithm 2, which we call `PD − BwK`. When we refer to the UCB or LCB for a latent parameter (the reward of an arm, or the amount of some resource that it utilizes), these are computed as follows. Letting $\hat{v}$ denote the empirical average of the observations of that random variable[3] and letting N denote the number of times the random variable has been observed, the lower confidence bound (LCB) and upper confidence bound (UCB) are the left and right endpoints, respectively, of the *confidence interval* $[0,1] \cap [\hat{v} - \texttt{rad}(\hat{v}, N),\ \hat{v} + \texttt{rad}(\hat{v}, N)]$. The UCB or LCB for a vector or matrix are defined component-wise.

In step 8, the pseudocode asserts that the set $\operatorname{argmin}_{z \in \mathbf{\Delta}[A]}\{\frac{y_t^\intercal L_t z}{u_t^\intercal z}\}$ contains at least one of the point-mass distributions $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$. This is true, because if $\rho = \min_{z \in \mathbf{\Delta}[A]}\{(y_t^\intercal L_t z)/(u_t^\intercal z)\}$ then the linear inequality $y_t^\intercal L_t z \leq \rho u_t^\intercal \mathcal{D}$ holds at some point $z \in \mathbf{\Delta}[A]$, and hence it holds at some extreme point, i.e. one of the point-

[3]Note that we initialize the algorithm by pulling each arm once, so empirical averages are always well-defined.

mass distributions.

Algorithm 2: Algorithm PD − BwK

1: Set $\epsilon = \sqrt{\ln(d)/B}$.

2: In the first m rounds, pull each arm once.

3: $v_1 = \mathbf{1}$

4: **for** $t = m+1, \ldots, \tau$ *(i.e., until resource budget exhausted)* **do**

5: Compute UCB estimate for reward vector, u_t.

6: Compute LCB estimate for resource consumption matrix, L_t.

7: $y_t = v_t/(\mathbf{1}^\intercal v_t)$.

8: Pull arm a_j from distribution z_t which minimizes $\frac{y_t^\intercal L_t z}{u_t^\intercal z}$.
(Note that $z_t = \mathbf{e}_j$ is point mass.)

9: $v_{t+1} = \text{Diag}\{(1+\epsilon)^{\mathbf{e}_j^\intercal L_t z_t}\} v_t$.

Another feature of our algorithm that deserves mention is a variant of the Garg-Könemann width reduction technique [29]. The ratio $(y_t^\intercal L_t z_t)/(u_t^\intercal z_t)$ that we optimize in step 8 may be unboundedly large, so in the multiplicative update in step 9 we rescale this value to $y_t^\intercal L_t z_t$, which is guaranteed to be at most 1. This rescaling is mirrored in the analysis of the algorithm, when we define the dual vector $\bar{y}$ by averaging the vectors y_t using the aforementioned scale factors. (Interestingly, unlike the Garg-Könemann algorithm which applies multiplicative updates to the dual vectors and weighted averaging to the primal ones, in our algorithm the multiplicative updates and weighted averaging are *both* applied to the dual vectors.)

The following theorem expresses the regret guarantee for PD − BwK. The proof is in Section 3.3.

Theorem 3.2.1. *For any instance of the* BwK *problem, the regret of Algorithm* PD − BwK *satisfies*

$$\texttt{OPT} - \texttt{REW} \leq O\left(\sqrt{\log(dmT)}\right)\left(\sqrt{m\,\texttt{OPT}} + \texttt{OPT}\,\sqrt{\frac{m}{B}} + m\sqrt{\log(dmT)}\right). \tag{3.6}$$

Comparing this bound to Theorem 3.4.1, when $d = O(1)$ and $M_{\texttt{LP}} = O(\texttt{OPT})$ the two guarantees are the same up to logarithmic factors. The regret guarantee for the primal-dual algorithm scales better with the number of resources, d, and it can be vastly superior in cases where $\texttt{OPT} \ll M_{\texttt{LP}}$.

3.3 Analysis of PD − BwK

In this section we present the analysis of the PD − BwK algorithm. We begin by recalling the Hedge algorithm from online learning theory, and its performance guarantee. Next, we present a simplified analysis of PD − BwK in a toy model in which the outcome vectors are deterministic. (This toy model of the BwK problem is uninteresting as a learning problem since the latent structure does not need to be learned, and the problem reduces to solving a linear program. We analyze PD − BwK first in this context because the analysis is simple, yet its main ideas carry over to the general case which is more complicated.)

3.3.1 Warm-up: The deterministic case

To present the application of Hedge to BwK in its purest form, we first present an algorithm for the case in which the rewards of the various arms are deterministically equal to the components of a vector $r \in \mathbb{R}^m$, and the resource consumption

vectors are deterministically equal to the columns of a matrix $C \in \mathbb{R}^{d \times m}$.

Algorithm 3: Algorithm PD − BwK, adapted for deterministic outcomes

1: In the first m rounds, pull each arm once.
2: Let r, C denote the reward vector and resource consumption matrix revealed in Step 1.
3: $v_1 = \mathbf{1}$
4: $t = 1$
5: **for** $t = m+1, \ldots, \tau$ *(i.e., until resource budget exhausted)* **do**
6: $\quad y_t = v_t/(\mathbf{1}^\intercal v_t)$.
7: $\quad$ Pull arm a_j from distribution z_t which minimizes $\frac{y_t^\intercal C z}{r^\intercal z}$.
$\quad$ (Note that $z_t = \mathbf{e}_j$ is point mass.)
8: $\quad v_{t+1} = \text{Diag}\{(1+\epsilon)^{\mathbf{e}_j^\intercal C z_t}\} v_t$.
9: $\quad t = t+1$.

This algorithm is an instance of the multiplicative-weights update method for solving packing linear programs. Interpreting it through the lens of online learning, as in the survey [3], it is updating a vector y using the Hedge algorithm. The payoff vector in round t is given by Ca_t. Note that these payoffs belong to $[0,1]$, by our assumption that C has $[0,1]$ entries.

Now let ξ^* denote an optimal solution of the primal linear program (P) from Section 3.1.1, and let $\texttt{OPT}_{\texttt{LP}} = r^\intercal \xi^*$ denote the optimal value of that LP. Let $\texttt{REW} = \sum_{t=m+1}^{\tau-1} r^\intercal z_t$ denote the total payoff obtained by the algorithm after its start-up phase (the first m rounds). By the stopping condition for the algorithm, we know there is a vector $y \in \Delta[\mathscr{R}]$ such that $y^\intercal C \left(\sum_{t=1}^{\tau} z_t\right) \geq B$ so fix one such y. Note that the algorithm's start-up phase consumes at most m units of any resource,

and the final round τ consumes at most one unit more, so

$$y^\intercal C\left(\sum_{m<t<\tau} z_t\right) \geq B - m - 1. \tag{3.7}$$

Finally let

$$\bar{y} = \frac{1}{\texttt{REW}} \sum_{m<t<\tau} (r^\intercal z_t) y_t.$$

Now we have

$$\begin{aligned} B \geq \bar{y}^\intercal C\xi^* &= \frac{1}{\texttt{REW}} \sum_{m<t<\tau} (r^\intercal z_t)(y_t^\intercal C\xi^*) \\ &\geq \frac{1}{\texttt{REW}} \sum_{m<t<\tau} (r^\intercal \xi^*)(y_t^\intercal C z_t) \\ &\geq \frac{\texttt{OPT}_{\texttt{LP}}}{\texttt{REW}} \left[(1-\epsilon) \sum_{m<t<\tau} y^\intercal C z_t - \frac{\ln d}{\epsilon}\right] \\ &\geq \frac{\texttt{OPT}_{\texttt{LP}}}{\texttt{REW}} \left[B - \epsilon B - m - 1 - \frac{\ln d}{\epsilon}\right]. \end{aligned}$$

The first inequality is by primal feasibility of ξ^*, the second is by the definition of z_t, the third is the performance guarantee of Hedge, the fourth is by (3.7). Putting these inequalities together we have derived

$$\frac{\texttt{REW}}{\texttt{OPT}_{\texttt{LP}}} \geq 1 - \epsilon - \frac{m+1}{B} - \frac{\ln d}{\epsilon B}.$$

Setting $\epsilon = \sqrt{\frac{\ln d}{B}}$ we find that the algorithm's regret is bounded above by $\texttt{OPT}_{\texttt{LP}} \cdot O\left(\sqrt{\frac{\ln d}{B}} + \frac{m}{B}\right)$.

3.3.2 Analysis modulo error terms

We now commence the analysis of Algorithm PD − BwK. In this section we show how to reduce the problem of bounding the algorithm's regret to a problem of estimating two error terms that reflect the difference between the algorithm's confidence-bound estimates of its own reward and resource consumption with

the empirical values of these random variables. The analysis of those error terms will be the subject of Section 3.3.3.

By Theorem 3.1.3 and our choice of $C_{\texttt{rad}}$, it holds with probability at least $1 - T^{-1}$ that the confidence interval for every latent parameter, in every round of execution, contains the true value of that latent parameter. We call this high-probability event a *clean execution* of $\texttt{PD} - \texttt{BwK}$. Our regret guarantee will hold deterministically assuming that a clean execution takes place. The regret can be at most T when a clean execution does not take place, and since this event has probability at most T^{-1} it contributes only $O(1)$ to the regret. We will henceforth assume a clean execution of $\texttt{PD} - \texttt{BwK}$.

Claim 3.3.1. *In a clean execution of Algorithm* $\texttt{PD} - \texttt{BwK}$*, the algorithm's total reward satisfies the bound*

$$\texttt{REW} \geq \texttt{OPT}_{\texttt{LP}} - \left[2\texttt{OPT}_{\texttt{LP}} \left(\sqrt{\frac{\ln d}{B}} + \frac{m+1}{B} \right) + m + 1 \right] - \frac{\texttt{OPT}_{\texttt{LP}}}{B} \left\| \sum_{m<t<\tau} E_t z_t \right\|_\infty - \left| \sum_{m<t<\tau} \delta_t^\intercal z_t \right|, \tag{3.8}$$

where E_t *and* δ_t *are defined by*

$$E_t = C_t - L_t, \qquad \delta_t = u_t - r_t. \tag{3.9}$$

Proof. The claim is proven by mimicking the analysis of Algorithm 3 in the preceding section, incorporating error terms that reflect the differences between observable values and latent ones. As before, let ξ^* denote an optimal solution of the primal linear program (P), and let $\texttt{OPT}_{\texttt{LP}} = r^\intercal \xi^*$ denote the optimal value of that LP. Let $\texttt{REW}_{\texttt{UCB}} = \sum_{m<t<\tau} u_t^\intercal z_t$ denote the total payoff the algorithm would have obtained, after its start-up phase, if the actual payoff at time t were replaced with the upper confidence bound. By the stopping condition for the algorithm, we know there is a vector $y \in \Delta[\mathscr{R}]$ such that $y^\intercal \left(\sum_{t=1}^{\tau} C_t z_t \right) \geq B$, so fix

one such y. As before,

$$y^\intercal \left(\sum_{m<t<\tau} C_t z_t \right) \geq B - m - 1. \tag{3.10}$$

Finally let

$$\bar{y} = \frac{1}{\texttt{REW}_{\texttt{UCB}}} \sum_{m<t<\tau} (u_t^\intercal z_t) y_t.$$

Assuming a clean execution, we have

$$\begin{aligned}
B &\geq \bar{y}^\intercal C \xi^* && (\xi^* \text{ is primal feasible}) \\
&= \frac{1}{\texttt{REW}_{\texttt{UCB}}} \sum_{m<t<\tau} (u_t^\intercal z_t)(y_t^\intercal C \xi^*) \\
&\geq \frac{1}{\texttt{REW}_{\texttt{UCB}}} \sum_{m<t<\tau} (u_t^\intercal z_t)(y_t^\intercal L_t \xi^*) && (\text{clean execution}) \\
&\geq \frac{1}{\texttt{REW}_{\texttt{UCB}}} \sum_{m<t<\tau} (u_t^\intercal \xi^*)(y_t^\intercal L_t z_t) && (\text{definition of } z_t) \\
&\geq \frac{1}{\texttt{REW}_{\texttt{UCB}}} \sum_{m<t<\tau} (r^\intercal \xi^*)(y_t^\intercal L_t z_t) && (\text{clean execution}) \\
&\geq \frac{\texttt{OPT}_{\texttt{LP}}}{\texttt{REW}_{\texttt{UCB}}} \left[(1-\epsilon) y^\intercal \left(\sum_{m<t<\tau} L_t z_t \right) - \frac{\ln d}{\epsilon} \right] && (\text{Hedge guarantee}) \\
&= \frac{\texttt{OPT}_{\texttt{LP}}}{\texttt{REW}_{\texttt{UCB}}} \left[(1-\epsilon) y^\intercal \left(\sum_{m<t<\tau} C_t z_t \right) - (1-\epsilon) y^\intercal \left(\sum_{m<t<\tau} E_t z_t \right) - \frac{\ln d}{\epsilon} \right] \\
&\geq \frac{\texttt{OPT}_{\texttt{LP}}}{\texttt{REW}_{\texttt{UCB}}} \left[(1-\epsilon) B - m - 1 - (1-\epsilon) y^\intercal \left(\sum_{m<t<\tau} E_t z_t \right) - \frac{\ln d}{\epsilon} \right] && (\text{definition of } y\text{; see eq. (3.10)}) \\
\texttt{REW}_{\texttt{UCB}} &\geq \texttt{OPT}_{\texttt{LP}} \left[1 - \epsilon - \frac{m+1}{B} - \frac{1}{B} \left\| \sum_{m<t<\tau} E_t z_t \right\|_\infty - \frac{\ln d}{\epsilon B} \right]. && (3.11)
\end{aligned}$$

The algorithm's actual payoff, $\texttt{REW} = \sum_{t=1}^{\tau} r_t^\intercal z_t$, satisfies the inequality

$$\texttt{REW} \geq \texttt{REW}_{\texttt{UCB}} - \sum_{m<t<\tau} (u_t - r_t)^\intercal z_t = \texttt{REW}_{\texttt{UCB}} - \sum_{m<t<\tau} \delta_\tau^\intercal z_t.$$

Combining this inequality with (3.11), we obtain the bound (3.8), as claimed. □

3.3.3 Error analysis

In this section we complete the proof of Theorem 3.2.1 by deriving upper bounds on the terms $\left\|\sum_{m<t<\tau} E_t z_t\right\|_\infty$ and $\left|\sum_{m<t<\tau} \delta_t z_t\right|$ appearing on the right side of (3.8). Both bounds are special cases of a more general statement, presented as Lemma 3.3.3 below. Before stating the lemma, we need to establish a simple fact about confidence radii.

Lemma 3.3.2. *For any two vectors $w, N \in \mathbb{R}^m_+$, we have*

$$\sum_{j=1}^{m} \mathtt{rad}(w_j, N_j)\, N_j \le \sqrt{C_{\mathtt{rad}}\, m(w^\intercal N)} + C_{\mathtt{rad}}\, m. \tag{3.12}$$

Proof. The definition of $\mathtt{rad}(\cdot,\cdot)$ implies that $\mathtt{rad}(w_j, N_j)\, N_j \le \sqrt{C_{\mathtt{rad}}\, w_j N_j} + C_{\mathtt{rad}}$. Summing these inequalities and applying Cauchy-Schwarz,

$$\sum_{j=1}^{m} \mathtt{rad}(w_j, N_j)\, N_j \le \sum_{j=1}^{m} \sqrt{C_{\mathtt{rad}}\, w_j N_j} + C_{\mathtt{rad}}\, m \le \sqrt{m} \cdot \sqrt{\sum_{j \in A} C_{\mathtt{rad}}\, w_j N_j} + C_{\mathtt{rad}}\, m,$$

and the lemma follows by rewriting the expression on the right side. □

The next lemma requires some notation and definitions. Suppose that $w_1, \ldots, w_\tau$ is a sequence of vectors in $[0,1]^m$ and that $z_1, \ldots, z_\tau$ is a sequence of vectors in $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$. Assume that the latter sequence begins with a permutation of the elements of $\{\mathbf{e}_1, \ldots, \mathbf{e}_m\}$ so that, for every $s \ge m$, the diagonal matrix $\sum_{t=1}^{s} z_t z_t^\intercal$ is invertible. The vector

$$\overline{w}_s = \left(\sum_{t=1}^{s} z_t z_t^\intercal\right)^{-1} \left(\sum_{t=1}^{s} z_t z_t^\intercal w_t\right)$$

can be interpreted as the vector of empirical averages: the entry $\overline{w}_{s,j}$ is equal to the average of $w_{t,j}$ over all times $t \le s$ such that $z_t = \mathbf{e}_j$. Let $\mathbf{N}_s = \sum_{t=1}^{s} z_t$ denote the

vector whose entry $\mathbf{N}_{s,j}$ indicates the number of times $\mathbf{e}_j$ occurs in the sequence $z_1, \ldots, z_s$. A useful identity is

$$\overline{w}_s{}^\intercal \mathbf{N}_t = \sum_{s=1}^{t} w_s^\intercal z_s.$$

Lemma 3.3.3. *Suppose we are given sequences of vectors $w_1, \ldots, w_\tau$ and $z_1, \ldots, z_\tau$ as above. Suppose we are additionally a sequence of vectors $b_1, \ldots, b_\tau$ in $[0,1]^m$ and another vector w_0 such that for $m < t < \tau$ and $j \in A$,*

$$|b_{t,j} - w_{0,j}| \le 2\,\mathtt{rad}(\overline{w}_{t,j}, N_{t,j}) \le 6\,\mathtt{rad}(w_{0,j}, N_{t,j}).$$

Let $s = \tau - 1$ and suppose that for all $j \in A$, $|\overline{w}_{s,j} - w_{0,j}| \le \mathtt{rad}(\overline{w}_{s,j}, N_{s,j})$. Then

$$\left| \sum_{m<t<\tau} (b_t - w_t)^\intercal z_t \right| \le O\left(\sqrt{C_{\mathtt{rad}}\, mW} + C_{\mathtt{rad}}\, m \right), \tag{3.13}$$

where $W = (\overline{w}_{\tau-1})^\intercal \mathbf{N}_{\tau-1} = \sum_{t=1}^{\tau-1} w_t^\intercal z_t$.

Proof. The proof decomposes the left side of (3.13) as a sum of three terms,

$$\sum_{m<t<\tau} (b_t - w_t)^\intercal z_t = \sum_{t=1}^{m} (w_t - b_t)^\intercal z_t + \sum_{t=1}^{\tau-1} (b_t - w_0)^\intercal z_t + \sum_{t=1}^{\tau-1} (w_0 - w_t)^\intercal z_t, \tag{3.14}$$

then bounds the three terms separately. The first sum is clearly bounded above by m. We next work on bounding the third sum. Let $s = \tau - 1$.

$$\begin{aligned}
\sum_{t=1}^{s} (w_0 - w_t)^\intercal z_t = w_0^\intercal \mathbf{N}_t - \sum_{t=1}^{s} w_t^\intercal z_t &= (w_0 - \overline{w}_s)^\intercal \mathbf{N}_s \\
\left| \sum_{t=1}^{s} (w_0 - w_t)^\intercal z_t \right| = |(w_0 - \overline{w}_s)^\intercal \mathbf{N}_s| &\le \sum_{j \in A} \mathtt{rad}(\overline{w}_{s,j}, N_{s,j}) N_{s,j} \\
&\le \sqrt{C_{\mathtt{rad}}\, mW} + C_{\mathtt{rad}}\, m,
\end{aligned}$$

where the last line follows from Lemma 3.3.2.

Finally we bound the middle sum in (3.14).

$$\begin{aligned}\left|\sum_{t=1}^{s}(b_t - w_0)^\intercal z_t\right| &\le 6\sum_{t=1}^{s}\sum_{j\in A}\texttt{rad}(w_{0,j}, N_{t,j})z_{t,j}\\ &= 6\sum_{j\in A}\sum_{\ell=1}^{N_{s,j}}\texttt{rad}(w_{0,j},\ell)\\ &= O\left(\sum_{j\in A}\texttt{rad}(w_{0,j}, N_{s,j})N_{s,j}\right)\\ &\le O\left(\sqrt{C_{\texttt{rad}}\, n w_0^\intercal \mathbf{N}_s} + C_{\texttt{rad}}\, m\right).\end{aligned}$$

We would like to replace the expression $w_0^\intercal \mathbf{N}_s$ on the last line with the expression $\overline{w}_s{}^\intercal \mathbf{N}_s = W$. To do so, recall from earlier in this proof that $|(w_0 - \overline{w}_s)^\intercal \mathbf{N}_s| \le \sqrt{C_{\texttt{rad}}\, mW} + C_{\texttt{rad}}\, m$, and now apply the following calculation:

$$w_0^\intercal \mathbf{N}_s \le \overline{w}_s{}^\intercal \mathbf{N}_s + \sqrt{C_{\texttt{rad}}\, mW} + C_{\texttt{rad}}\, m = W + \sqrt{C_{\texttt{rad}}\, mW} + C_{\texttt{rad}}\, m \le \left(\sqrt{W} + \sqrt{C_{\texttt{rad}}\, m}\right)^2$$

$$\sqrt{C_{\texttt{rad}}\, m w_0^\intercal \mathbf{N}_s} \le \sqrt{C_{\texttt{rad}}\, m}\left(\sqrt{W} + \sqrt{C_{\texttt{rad}}\, m}\right) = \sqrt{C_{\texttt{rad}}\, mW} + C_{\texttt{rad}}\, m.$$

Summing up the upper bounds for the three terms on the right side of (3.14), we obtain (3.13). □

Corollary 3.3.4. *In a clean execution of* PD − BwK*,*

$$\left|\sum_{m<t<\tau}\delta_t z_t\right| \le O\left(\sqrt{C_{\texttt{rad}}\, m\texttt{REW}} + C_{\texttt{rad}}\, m\right)$$

and

$$\left\|\sum_{m<t<\tau} E_t z_t\right\|_\infty \le O\left(\sqrt{C_{\texttt{rad}}\, mB} + C_{\texttt{rad}}\, m\right).$$

Proof. The first inequality is obtained by applying Lemma 3.3.3 with vector sequences $w_t = r_t$ and $b_t = u_t$. The second one is obtained by applying the same lemma with vector sequences $w_t = \mathbf{e}_i^\intercal C_t$ and $b_t = \mathbf{e}_i L_t$, for each resource i. □

Proof of Theorem 3.2.1: The theorem asserts that

$$\texttt{REW} \geq \texttt{OPT} - O\left(\sqrt{m\log(dmT)}\left(\frac{\texttt{OPT}}{\sqrt{B}} + \sqrt{\texttt{OPT}} + \sqrt{m\log(dmT)}\right)\right).$$

Denote the right-hand side by $f(\texttt{OPT})$. The function $\max\{f(x), 0\}$ is a non-decreasing function of x, so to establish the theorem we will prove that $\texttt{REW} \geq f(\texttt{OPT}_{\texttt{LP}}) \geq f(\texttt{OPT})$, where the latter inequality is an immediate consequence of the fact that $\texttt{OPT}_{\texttt{LP}} \geq \texttt{OPT}$ (Lemma 3.1.1).

To prove $\texttt{REW} \geq f(\texttt{OPT}_{\texttt{LP}})$, we begin by recalling inequality (3.8) and observing that

$$2\texttt{OPT}_{\texttt{LP}}\left(\sqrt{\frac{\ln d}{B}} + \frac{m+1}{B}\right) = O\left(\sqrt{m\log(dmT)}\,\frac{\texttt{OPT}_{\texttt{LP}}}{\sqrt{B}}\right)$$

by our assumption that $m \leq B$. The term $m + 1$ on the right side of Equation (3.8) is bounded above by $m\log(dmT)$. Finally, using Corollary 3.3.4 we see that the sum of the final two terms on the right side of (3.8) is bounded by $O\left(\sqrt{C_{\texttt{rad}}\,m}\left(\frac{\texttt{OPT}_{\texttt{LP}}}{\sqrt{B}} + \sqrt{\texttt{OPT}_{\texttt{LP}}} + \sqrt{C_{\texttt{rad}}\,m}\right)\right)$. The theorem follows by plugging in $C_{\texttt{rad}} = \Theta(\log(dmT))$. □

3.4 Algorithm: `Mixture_Elimintation`

The design principle behind `Mixture_Elimintation` is to explore as much as possible while avoiding obviously suboptimal strategies. On a high level, the algorithm is very simple. The goal is to converge on an LP-perfect distribution. The time is divided in phases of $|A|$ rounds each. In the beginning of each phase p, the algorithm prunes away all distributions $\mathcal{D}$ over arms that with high confidence are not LP-perfect given the observations so far. The remaining distributions over arms are called *potentially perfect*. Throughout the phase, the

algorithm chooses among the potentially perfect distributions. Specifically, for each arm a, the algorithm chooses a potentially perfect distribution $\mathcal{D}_{p,a}$ which approximately maximizes $\mathcal{D}_{p,a}(a)$, and "pulls" an arm sampled independently from this distribution. This choice of $\mathcal{D}_{p,a}$ is crucial; we call it the *balancing step*. The algorithm halts as soon as the time horizon is met, or any of the constraints is exhausted. See Algorithm 4 for the pseudocode.

Algorithm 4: `Mixture_Elimintation`

For each phase $p = 0, 1, 2, \ldots$ **do**
  Recompute the set $\mathbf{\Delta}_p$ of potentially perfect distributions $\mathcal{D}$ over arms.
  Over the next $|A|$ rounds, for each $a \in A$:
    pick any distribution $\mathcal{D} = \mathcal{D}_{p,a} \in \mathbf{\Delta}_p$ such that $\mathcal{D}(a) \geq \frac{1}{2} \max_{\mathcal{D} \in \mathbf{\Delta}_p} \mathcal{D}(a)$.
    choose an arm to "pull" as an independent sample from $\mathcal{D}$.
    halt if time horizon is met or one of the resources is exhausted.

We believe that `Mixture_Elimintation`, like `UCB1` [4], is a very general design principle and has the potential to be a meta-algorithm for solving stochastic online learning problems.

Theorem 3.4.1 (`BwK`: `Mixture_Elimintation`). *Consider an instance of* `BwK` *with d resources, $m = |A|$ arms, and the smallest budget $B = \min_i B_i$. Let $M_{\texttt{LP}}$ be the maximum of $\texttt{OPT}_{\texttt{LP}}$, for given time horizon and budgets, over all problem instances in a given BwK domain. Algorithm* `Mixture_Elimintation` *achieves total expected regret*

$$\texttt{OPT} - \texttt{REW} \leq O(\log(dmT)\log(T/m)) \left(\sqrt{dm\, M_{\texttt{LP}}} + M_{\texttt{LP}} \left(\sqrt{\frac{dm}{B}} + \frac{dm}{B} \right) + dm \right). \quad (3.15)$$

Let us fill in the details in the specification of `Mixture_Elimintation`. In the beginning of each phase p, the algorithm recomputes a "confidence interval"

I_p for the latent structure μ, so that (informally) $\mu \in I_p$ with high probability. Then the algorithm determines which distributions $\mathcal{D}$ over arms can potentially be LP-perfect given that $\mu \in I_p$. Specifically, let $\mathbf{\Delta}_p$ be set of all distributions $\mathcal{D}$ that are LP-perfect for some latent structure $\mu' \in I_p$; such distributions are called *potentially perfect* (for phase p).

It remains to define the confidence intervals I_p. For phase $p = 0$, the confidence interval I_0 is simply $\mathcal{M}_{\texttt{feas}}$, the set of all feasible latent structures. For each subsequent phase $p \geq 1$, the confidence interval I_p is defined as follows. For each arm a, consider all rounds before phase p in which this arm has been chosen. Let $N_p(a)$ be the number of such rounds, let $\widehat{r}_p(a)$ be the time-averaged reward in these rounds, and let $\widehat{c}_{p,i}(a)$ be the time-averaged consumption of resource i in these rounds. We use these averages to estimate $r(a,\mu)$ and $c_i(a,\mu)$ as follows:

$$|r(a,\mu) - \widehat{r}_p(a)| \leq \texttt{rad}\Big(\widehat{r}_p(a), N_p(a)\Big) \tag{3.16}$$

$$|c_i(a,\mu) - \widehat{c}_{p,i}(a)| \leq \texttt{rad}\Big(\widehat{c}_{p,i}(a), N_p(a)\Big) \quad \text{for each resource } i \tag{3.17}$$

The confidence interval I_p is the set of all latent structures $\mu' \in I_{p-1}$ that are consistent with these estimates. This completes the specification of `Mixture_Elimintation`.

For each phase of `Mixture_Elimintation`, the round in which an arm is sampled from distribution $\mathcal{D}_{p,a}$ will be called *designated* to arm a. We need to use approximate maximization to choose $\mathcal{D}_{p,a}$, rather than exact maximization, because an exact maximizer $\operatorname{argmax}_{\mathcal{D}\in\mathbf{\Delta}_p} \mathcal{D}(a)$ is not guaranteed to exist.

3.5 Analysis of `Mixture_Elimintation`

Let us summarize various properties of the confidence radius which we use throughout the analysis.

Claim 3.5.1. *The confidence radius* $\texttt{rad}(\nu, N)$*, defined in Equation (3.3), satisfies the following properties:*

(a) monotonicity: $\texttt{rad}(\nu, N)$ *is non-decreasing in* ν *and non-increasing in* N*.*

(b) concavity: $\texttt{rad}(\nu, N)$ *is concave in* ν*, for any fixed* N*.*

(c) $\max(0,\ \nu - \texttt{rad}(\nu, N))$ *is non-decreasing in* ν*.*

(d) $\nu - \texttt{rad}(\nu, N) \geq \frac{1}{4}\,\nu$ *whenever* $4\frac{C_{\texttt{rad}}}{N} \leq \nu \leq 1$*.*

(e) $\texttt{rad}(\nu, N) \leq 3\frac{C_{\texttt{rad}}}{N}$ *whenever* $\nu \leq 4\frac{C_{\texttt{rad}}}{N}$*.*

(f) $\texttt{rad}(\nu, \alpha N) = \frac{1}{\alpha}\,\texttt{rad}(\alpha\nu,\ N)$*, for any* $\alpha \in (0, 1]$*.*

(g) $\frac{1}{N}\sum_{\ell=1}^{N} \texttt{rad}(\nu, \ell) \leq O(\log N)\ \texttt{rad}(\nu, N)$*.*

3.5.1 Deterministic properties of `Mixture_Elimintation`

First, we show that any two latent structures in the confidence interval I_p correspond to similar consumptions and rewards, for each arm a. This follows deterministically from the specification of I_p.

Claim 3.5.2. *Fix any phase* p*, any two latent structures* $\mu', \mu'' \in I_p$*, an arm* a*, and a resource* i*. Then*

$$|c_i(a, \mu') - c_i(a, \mu'')| \leq 6\ \texttt{rad}\Big(c_i(a, \mu'), N_p(a)\Big) \tag{3.18}$$

$$|r(a, \mu') - r(a, \mu'')| \leq 6\ \texttt{rad}\Big(r(a, \mu'), N_p(a)\Big). \tag{3.19}$$

Proof. We prove Equation (3.18); Equation (3.19) is proved similarly.

Let $N = N_p(a)$. By specification of `Mixture_Elimintation`, any $\mu' \in I_p$ is consistent with estimate (4.5):

$$|c_i(a,\mu') - \widehat{c}_{p,i}(a)| \leq \texttt{rad}\left(\widehat{c}_{p,i}(a), N\right).$$

It follows that

$$|c_i(a,\mu') - c_i(a,\mu'')| \leq 2\ \texttt{rad}\left(\widehat{c}_{p,i}(a), N\right).$$

Finally, we observe that by Claim 3.1.4,

$$\texttt{rad}\left(\widehat{c}_{p,i}(a), N\right) \leq 3\,\texttt{rad}\left(c_i(a,\mu'), N\right). \quad \square$$

For each phase p and arm a, let $\bar{\mathcal{D}}_{p,a} = \frac{1}{p}\sum_{q<p} \mathcal{D}_{q,a}(a)$ be the average of probabilities for arm a among the distributions in the preceding phases that are designated to arm a. Because of the balancing step in `Mixture_Elimintation`, we can compare this quantity to $\mathcal{D}(a)$, for any $\mathcal{D} \in \mathbf{\Delta}_p$. (Here we also use the fact that the confidence intervals I_p are non-increasing from one phase to another.)

Claim 3.5.3. *$\bar{\mathcal{D}}_{p,a} \geq \frac{1}{2}\mathcal{D}(a)$ for each phase p, each arm a and any distribution $\mathcal{D} \in \mathbf{\Delta}_p$.*

Proof. Fix arm a. Recall that $\bar{\mathcal{D}}_{p,a} = \frac{1}{p}\sum_{q<p} \mathcal{D}_{q,a}(a)$, where $\mathcal{D}_{q,a}$ is the distribution chosen in the round in phase q that is designated to arm a. Fix any phase $q < p$. Because of the balancing step, $\mathcal{D}_{q,a}(a) \geq \frac{1}{2}\mathcal{D}'(a)$ for any distribution $\mathcal{D}' \in \mathbf{\Delta}_q$. Since the confidence intervals I_q are non-increasing from one phase to another, we have $I_p \subset I_q$ for any $q \leq p$, which implies that $\mathbf{\Delta}_p \subset \mathbf{\Delta}_q$. Consequently, $\mathcal{D}_{q,a}(a) \geq \frac{1}{2}\mathcal{D}(a)$ for each $q < p$, and the claim follows. $\square$

3.5.2 High-probability events

We keep track of several quantities: the averages $\widehat{r}_p(a)$ and $\widehat{c}_{p,i}(a)$ defined in Section 3.4, as well as several other quantities that we define below.

Fix phase p. For each arm a, consider all rounds in phases $q < p$ that are designated to arm a. Let $n_p(a)$ denote the number of times arm a has been chosen in these rounds. Let $\widehat{\mathcal{D}}_{p,a} = n_t(a)/p$ be the corresponding empirical probability of choosing x. We compare this to $\bar{\mathcal{D}}_{p,a}$.

Further, consider all rounds in phases $q < p$. There are $N = p|A|$ such rounds. The average distribution chosen by the algorithm in these rounds is $\bar{\mathcal{D}}_p = \frac{1}{N} \sum_{q<p,\, a\in A} \mathcal{D}_{q,a}$. We are interested in the corresponding quantities $r(\bar{\mathcal{D}}_p, \mu)$ and $c_i(\bar{\mathcal{D}}_p, \mu)$, We compare these quantities to $\widehat{r}_p = \frac{1}{N} \sum_{t=1}^{N} r_t$ and $\widehat{c}_{p,i} = \frac{1}{N} \sum_{t=1}^{N} c_{t,i}$, the average reward and the average resource-i consumption in phases $q < p$.

We consider several high-probability events which follow from applying Theorem 3.1.3 to the various quantities defined above. All these events have a common shape: $\widehat{\nu}$ is an N-strong estimate for ν, for some quantities $\nu, \widehat{\nu} \in [0, 1]$ and $N \in \mathbb{N}$.

Lemma 3.5.4. *For each phase p, arm a, and resource i, with probability $e^{-\Omega(C_{\text{rad}})}$ it holds that:*

(a) $\widehat{r}_p(a)$ is an $N_p(a)$-strong estimator for $r(a, \mu)$, and $\widehat{c}_{p,i}(a)$ is an $N_p(a)$-strong estimator for $c_i(a, \mu)$.

(b) $\bar{\mathcal{D}}_{p,a}$ is an p-strong estimator for $\widehat{\mathcal{D}}_{p,a}$.

(c) $r(\bar{\mathcal{D}}_p, \mu)$ is an $(p|A|)$-strong estimator for $\widehat{r}_p$, and $c_i(\bar{\mathcal{D}}_p, \mu)$ is an $(p|A|)$-strong estimator for $\widehat{c}_{p,i}$.

3.5.3 Clean execution of `Mixture_Elimintation`

A *clean execution* of the algorithm is one in which all events in Lemma 3.5.4 hold. It is convenient to focus on a clean execution. In the rest of the analysis, we assume clean execution without further notice. Also, we fix an arbitrary phase p in such execution.

Since a clean execution satisfies the event in Claim 3.5.4(a), it immediately follows that:

Claim 3.5.5. *The confidence interval I_p contains the (actual) latent structure μ. Therefore, $\mathcal{D}^* \in \mathbf{\Delta}_p$ for any distribution $\mathcal{D}^*$ that is LP-perfect for μ.*

Claim 3.5.6. *Fix any latent structures $\mu', \mu'' \in I_p$ and any distribution $\mathcal{D} \in \mathbf{\Delta}_p$. Then for each resource i,*

$$|c_i(\mathcal{D}, \mu') - c_i(\mathcal{D}, \mu'')| \leq O(1)\ \texttt{rad}\left(c_i(\mathcal{D}, \mu'),\ p/d\right) \tag{3.20}$$

$$|r(\mathcal{D}, \mu') - r(\mathcal{D}, \mu'')| \leq O(1)\ \texttt{rad}\left(r(\mathcal{D}, \mu'),\ p/d\right). \tag{3.21}$$

Proof. We prove Equation (3.20); Equation (3.21) is proved similarly. Let us first prove the following:

$$\forall a \in A, \quad \mathcal{D}(a)\,|c_i(a, \mu') - c_i(a, \mu'')| \leq O(1)\ \texttt{rad}(\mathcal{D}(a)\,c_i(a, \mu'), p). \tag{3.22}$$

Intuitively, in order to argue that we have good estimates on quantities related to arm a, it helps to prove that this arm has been chosen sufficiently often. Using the definition of clean execution and Claim 3.5.3, we accomplish this as follows:

$$\begin{aligned}
\tfrac{1}{p}\,N_p(a) &\geq \tfrac{1}{p}\,n_p(a) = \widehat{\mathcal{D}}_{p,a} \\
&\geq \bar{\mathcal{D}}_{p,a} - \texttt{rad}(\bar{\mathcal{D}}_{p,a}, p) && \text{(by clean execution)} \\
&\geq \tfrac{1}{2}\,\mathcal{D}(a) - \texttt{rad}(\tfrac{1}{2}\,\mathcal{D}(a), p) && \text{(by Claim 3.5.3 and Claim 3.5.1(c)).}
\end{aligned}$$

Consider two cases depending on $\mathcal{D}(a)$. For the first case, assume $\mathcal{D}(a) \geq 8\frac{C_{\mathtt{rad}}}{p}$. Using Claim 3.5.1(d) and the previous equation, it follows that $N_p(a) \geq \frac{1}{8}\, p\, \mathcal{D}(a)$. Therefore:

$$\begin{aligned}
\mathcal{D}(a)\, |c_i(a,\mu') - c_i(a,\mu'')| &\leq 6\, \mathcal{D}(a)\, \mathtt{rad}(c_i(a,\mu'),\, N_p(a)) && \text{(by Claim 3.5.2)} \\
&\leq 6\, \mathcal{D}(a)\, \mathtt{rad}(c_i(a,\mu'),\, \tfrac{1}{8}\, p\, \mathcal{D}(a)) && \text{(by monotonicity of } \mathtt{rad}) \\
&= 48\, \mathtt{rad}(\mathcal{D}(a)\, c_i(a,\mu'_a), p) && \text{(by Claim 3.5.1(f)).}
\end{aligned}$$

The second case is that $\mathcal{D}(a) < 8\frac{C_{\mathtt{rad}}}{p}$. Then Equation (3.22) follows simply because $\frac{C_{\mathtt{rad}}}{p} \leq \mathtt{rad}(\cdot\,, p)$.

We have proved Equation (3.22). We complete the proof of Equation (3.20) using concavity of $\mathtt{rad}(\cdot, p)$ and the fact that, by the specification of `Mixture_Elimintation`, $\mathcal{D}$ has support of size at most $d+1$.

$$\begin{aligned}
|c_i(\mathcal{D},\mu') - c_i(\mathcal{D},\mu'')| &\leq \textstyle\sum_{a\in A} \mathcal{D}(a)\, |c_i(a,\mu') - c_i(a,\mu'')| \\
&\leq \textstyle\sum_{a\in A, \mathcal{D}(a)>0} O(1)\, \mathtt{rad}(\mathcal{D}(a)\, c_i(a,\mu'), p) \\
&\leq O(d)\, \mathtt{rad}\Big(\tfrac{1}{d+1}\, \textstyle\sum_{a\in A} \mathcal{D}(a)\, c_i(a,\mu'),\ p\Big) \\
&= O(d)\, \mathtt{rad}\Big(\tfrac{1}{d+1}\, c_i(\mathcal{D},\mu'),\ p\Big) \\
&\leq O(1)\, \mathtt{rad}\Big(c_i(\mathcal{D},\mu'),\ \tfrac{p}{d}\Big) \qquad \text{(by Claim 3.5.1(f)).} \qquad \square
\end{aligned}$$

Claim 3.5.7. *Fix any latent structures $\mu', \mu'' \in I_p$ and any distribution $\mathcal{D} \in \mathbf{\Delta}_p$. Then*

$$|\mathtt{LP}(\mathcal{D},\mu') - \mathtt{LP}(\mathcal{D},\mu'')| \leq O(T)\, \mathtt{rad}\Big(M_{\mathtt{LP}}/T,\ \tfrac{p}{d}\Big) + O(M_{\mathtt{LP}}\, \tfrac{T}{B})\, \mathtt{rad}\Big(\tfrac{B}{T}, \tfrac{p}{d}\Big). \qquad (3.23)$$

Proof. Since $\mathcal{D} \in \mathbf{\Delta}_p$, it is LP-perfect for some latent structure μ. Then $\mathtt{LP}(\mathcal{D},\mu) = T\, r(\mathcal{D},\mu)$. Therefore:

$$\begin{aligned}
\mathtt{LP}(\mathcal{D},\mu') - \mathtt{LP}(\mathcal{D},\mu) &\leq\ T\ (r(\mathcal{D},\mu') - r(\mathcal{D},\mu)) \\
&\leq\ O(T)\, \mathtt{rad}\Big(r(\mathcal{D},\mu),\, \tfrac{p}{d}\Big) \qquad \text{(by Claim 3.5.6).} \qquad (3.24)
\end{aligned}$$

We need a little more work to bound the difference in the LP values in the other direction.

Consider $t_0 = \texttt{LP}(\mathcal{D}, \mu')/r(\mathcal{D}, \mu')$; this is the optimal value of variable t in linear program (3.1). Let us obtain a lower bound on this quantity. Assume $t_0 < T$. Then one of the budget constraints in (3.1) must be tight, i.e. $t_0\, c_i(\mathcal{D}, \mu') = B$ for some resource i.

$$\begin{aligned} c_i(\mathcal{D}, \mu') &\leq c_i(\mathcal{D}, \mu) + O(1)\, \texttt{rad}\left(c_i(\mathcal{D}, \mu), \tfrac{p}{d}\right) && \text{(by Claim 3.5.6)} \\ &\leq \tfrac{B}{T} + O(1)\, \texttt{rad}\left(\tfrac{B}{T}, \tfrac{p}{d}\right) \end{aligned}$$

Let $\Psi = \texttt{rad}\left(\frac{B}{T}, \frac{p}{d}\right)$. It follows that $t_0 = B/c_i(\mathcal{D}, \mu') \geq T(1 - O(\frac{T}{B}\Psi))$. Therefore:

$$\begin{aligned} \texttt{LP}(\mathcal{D}, \mu) - \texttt{LP}(\mathcal{D}, \mu') = {} & T\, r(\mathcal{D}, \mu) - t_0\, r(\mathcal{D}, \mu') \\ &\leq T\, r(\mathcal{D}, \mu) - \left[T(1 - O(\tfrac{T}{B}\Psi))\right]\, r(\mathcal{D}, \mu') \\ &\leq T\, \left[r(\mathcal{D}, \mu) - r(\mathcal{D}, \mu')\right] + O(\tfrac{T}{B}\Psi)\, T\, r(\mathcal{D}, \mu) \\ &\leq O(T)\, \texttt{rad}\left(r(\mathcal{D}, \mu), \tfrac{p}{d}\right) + O(\tfrac{T}{B}\Psi)\, T\, r(\mathcal{D}, \mu) \quad \text{(by Claim 3.5.6).} \end{aligned}$$

Using Equation (3.24) and noting that $r(\mathcal{D}, \mu) = \texttt{LP}(\mathcal{D}, \mu)/T \leq M_{\texttt{LP}}/T$, we conclude that

$$|\texttt{LP}(\mathcal{D}, \mu) - \texttt{LP}(\mathcal{D}, \mu')| \leq O(T)\, \texttt{rad}\left(M_{\texttt{LP}}/T,\ \tfrac{p}{d}\right) + O(M_{\texttt{LP}}\, \tfrac{T}{B})\, \texttt{rad}\left(\tfrac{B}{T}, \tfrac{p}{d}\right).$$

We obtain the same upper bound on $|\texttt{LP}(\mathcal{D}, \mu) - \texttt{LP}(\mathcal{D}, \mu'')|$, and the claim follows.

□

We will carry the upper bound in Equation (3.23) through the rest of the analysis. To simplify the equations, we will denote it Φ_p.

Claim 3.5.8. *Let Φ_p denote the right-hand side of Equation (3.23).*

(a) Fix any latent structure $\mu^* \in I_p$, and any distributions $\mathcal{D}', \mathcal{D}'' \in \mathbf{\Delta}_p$. Then

$$|\texttt{LP}(\mathcal{D}', \mu^*) - \texttt{LP}(\mathcal{D}'', \mu^*)| \leq 2\Phi_p.$$

(b) $\texttt{LP}(\mathcal{D}_{p,a}, \mu) \geq \texttt{OPT}_{\texttt{LP}} - 2\Phi_p$, for each distribution $\mathcal{D}_{p,a}$ chosen by the algorithm in phase p.

Proof. **(a).** Since $\mathcal{D}', \mathcal{D}'' \in \mathbf{\Delta}_p$, it holds that $\mathcal{D}'$ and $\mathcal{D}''$ are LP-perfect for some latent structures μ' and μ''. Further, pick a distribution $\mathcal{D}^*$ that is LP-perfect for μ^*. Then:

$$\begin{aligned}
\texttt{LP}(\mathcal{D}', \mu^*) &\geq \texttt{LP}(\mathcal{D}', \mu') - \Phi_p && \text{(by Lemma 3.5.7 with } \mathcal{D} = \mathcal{D}') \\
&\geq \texttt{LP}(\mathcal{D}^*, \mu') - \Phi_p \\
&\geq \texttt{LP}(\mathcal{D}^*, \mu^*) - 2\Phi_p && \text{(by Lemma 3.5.7 with } \mathcal{D} = \mathcal{D}^*) \\
&\geq \texttt{LP}(\mathcal{D}'', \mu^*) - 2\Phi_p. && \square
\end{aligned}$$

(b). Follows from part (a) because $\mathcal{D}_{p,a} \in \mathbf{\Delta}_p$ by the specification of `Mixture_Elimintation`, and $\texttt{OPT}_{\texttt{LP}} = \texttt{LP}(\mathcal{D}^*, \mu)$ for some distribution $\mathcal{D}^* \in \mathbf{\Delta}_p$.

The following corollary lower-bounds the average reward; once we have it, it essentially remains to lower-bound the stopping time of the algorithm.

Corollary 3.5.9. *$\widehat{r}_p \geq \frac{1}{T}\left(\texttt{OPT}_{\texttt{LP}} - O(\Phi_p \log p)\right)$, where Φ_p denotes the right-hand side of Equation (3.23).*

Proof. By Claim 3.5.8(b), for each distribution $\mathcal{D}_{q,a}$ chosen by the algorithm in phase $q < p$ it holds that

$$r(\mathcal{D}_{q,a}, \mu) \geq \tfrac{1}{T}\,\texttt{LP}(\mathcal{D}_{q,a}, \mu) \geq \tfrac{1}{T}\left(\texttt{OPT}_{\texttt{LP}} - 2\Phi_q\right).$$

Averaging the above equation over all rounds in phases $q < p$, we obtain

$$\begin{aligned} r(\bar{\mathcal{D}}_p, \mu) &\geq \tfrac{1}{T}\left(\texttt{OPT}_{\texttt{LP}} - \tfrac{2}{p}\textstyle\sum_{q<p}\Phi_q\right) \\ &\geq \tfrac{1}{T}\left(\texttt{OPT}_{\texttt{LP}} - O(\Phi_p \log p)\right). \end{aligned}$$

For the last inequality, we used Claim 3.5.1(fg) to average the confidence radii in Φ_q.

Using the high-probability event in Claim 3.5.4(c):

$$\widehat{r}_p \geq r(\bar{\mathcal{D}}_p, \mu) - \texttt{rad}(r(\bar{\mathcal{D}}_p, \mu), p|A|).$$

Now using the monotonicity of $\nu - \texttt{rad}(\nu, N)$ (Claim 3.5.1(c)) we obtain

$$\begin{aligned} \widehat{r}_p &\geq \tfrac{1}{T}\left(\texttt{OPT}_{\texttt{LP}} - O(\Phi_p)\right) - \texttt{rad}\left(\ \tfrac{1}{T}\left(\texttt{OPT}_{\texttt{LP}} - O(\Phi_p)\right),\ p|A|\ \right) \\ &\geq \tfrac{1}{T}\left(\texttt{OPT}_{\texttt{LP}} - O(\Phi_p)\right) - \texttt{rad}\left(\ M_{\texttt{LP}}/T,\ p|A|\ \right) \\ &\geq \tfrac{1}{T}\left(\texttt{OPT}_{\texttt{LP}} - O(\Phi_p)\right). \end{aligned}$$

For the last equation, we use the fact that $\Phi_p/T \geq \Omega(\texttt{rad}(M_{\texttt{LP}}/T), \frac{p}{d})) \geq \Omega(\texttt{rad}(\ M_{\texttt{LP}}/T,\ p|A|\ ))$. □

The following two claims help us to lower-bound the stopping time of the algorithm.

Claim 3.5.10. *$c_i(\mathcal{D}_{p,a}, \mu) \leq \frac{B}{T} + O(1)\,\texttt{rad}\left(\frac{B}{T},\ \frac{p}{d}\right)$ for each resource i.*

Proof. By the algorithm's specification, $\mathcal{D}_{p,a} \in \mathbf{\Delta}_p$, and moreover there exists a latent structure $\mu' \in I_p$ such that $\mathcal{D}_{p,a}$ is LP-perfect for μ'. Apply Claim 3.5.6, noting that $c_i(\mathcal{D}_{p,a}, \mu') \leq \frac{B}{T}$ by LP-perfectness. □

Corollary 3.5.11. *$\widehat{c}_{p,i} \leq \frac{B}{T} + O(\log p)\ \texttt{rad}(\frac{B}{T}, \frac{p}{d})$ for each resource i.*

Proof. Using a property of the clean execution, namely the event in Claim 3.5.4(c), we have

$$\widehat{c}_{p,i} \le c_i(\bar{\mathcal{D}}, \mu) + \texttt{rad}\Big(c_i(\bar{\mathcal{D}}, \mu),\ p\Big). \tag{3.25}$$

Consider all rounds preceding phase p.

$$\begin{aligned} c_i(\bar{\mathcal{D}}_p, \mu) &= \frac{1}{p|A|} \sum_{q<p,\, a\in A} c_i(\mathcal{D}_{q,a}, \mu) \\ &\le \frac{B}{T} + \frac{O(1)}{p|A|} \sum_{q<p,\, a\in A} \texttt{rad}\left(\frac{B}{T}, \frac{p}{d}\right) && \text{(by Claim 3.5.10)} \\ &\le \tfrac{B}{T} + O(\log p)\, \texttt{rad}\left(\tfrac{B}{T}, \tfrac{p}{d}\right) && \text{(by Claim 3.5.1(fg)).} \end{aligned} \tag{3.26}$$

For the last inequality, we used Claim 3.5.1(fg) to average the confidence radii.

Using the upper bound on $c_i(\bar{\mathcal{D}}, \mu)$ that we derived above,

$$\texttt{rad}\Big(c_i(\bar{\mathcal{D}}, \mu),\ \tfrac{p}{d}\Big) \le O(\log p)\, \texttt{rad}\Big(\tfrac{B}{T} + \texttt{rad}\Big(\tfrac{B}{T}, \tfrac{p}{d}\Big),\ \tfrac{p}{d}\Big).$$

Using a general property of the confidence radius that

$$\texttt{rad}(\nu + \texttt{rad}(\nu, N),\ N) \le O(\texttt{rad}(\nu, N)),$$

we conclude that

$$\texttt{rad}\Big(c_i(\bar{\mathcal{D}}, \mu),\ \tfrac{p}{d}\Big) \le O(\log p)\, \texttt{rad}(\tfrac{B}{T}, \tfrac{p}{d}). \tag{3.27}$$

We obtain the claim by plugging the upper bounds (3.26) and (3.27) into (3.25).

□

We are ready to put the pieces together and derive the performance guarantee for a clean execution of `Mixture_Elimintation`.

Lemma 3.5.12. *Consider a clean execution of* `Mixture_Elimintation`*. Letting* Φ_p *denote the right-hand side of Equation (3.23), the total reward of the algorithm is* $\texttt{REW} \geq \texttt{OPT}_{\text{LP}} - O\left(\Phi_{T/|A|} \log(T/|A|)\right)$.

Proof. Let p be the last phase in the execution of the algorithm, and let T_0 be the stopping time. Letting $m = |A|$, note that $pm < T_0 \leq (p+1)m$.

We can use Corollary 3.5.9 to bound REW from below:

$$\texttt{REW} = T_0 \widehat{r}_{p+1} > p\, m \widehat{r}_{p+1} \geq \tfrac{p\,m}{T} \left(\texttt{OPT}_{\text{LP}} - O(\Phi_p \log p)\right). \tag{3.28}$$

Let us bound $\frac{p\,m}{T}$ from below. The algorithm stops either when it runs out of time or if it runs out of resources during phase p. In the former case, $p = \lfloor T/m \rfloor$. In the latter case, $B = T_0 \widehat{c}_{p+1,i}$ for some resource i, so $B \leq m(p+1)\, \widehat{c}_{p+1,i}$. Using Corollary 3.5.11, we obtain the following lower bound on p:

$$\frac{p\,m}{T} \geq 1 - O\left(\frac{p\,m \log p}{B}\right) \texttt{rad}\left(\frac{B}{T}, \frac{p}{d}\right).$$

Plugging this into Equation (3.28), we conclude:

$$\begin{aligned}
\texttt{REW} &\geq \texttt{OPT}_{\text{LP}} - O\left(\frac{p\,m \log p}{B}\right) \texttt{rad}\left(\frac{B}{T}, \frac{p}{d}\right) \texttt{OPT}_{\text{LP}} - O\left(\frac{p\,m \log p}{T}\right) \Phi_p \\
&\geq \texttt{OPT}_{\text{LP}} - O\left(\frac{p\,m \log p}{T}\right) \left(\Phi_p + \frac{T}{B} \texttt{rad}\left(\frac{B}{T}, \frac{p}{d}\right) \texttt{OPT}_{\text{LP}}\right) \\
&\geq \texttt{OPT}_{\text{LP}} - \frac{p\,m \log p}{T}\, O(\Phi_p).
\end{aligned}$$

To complete the proof, we observe that $(p\,\Phi_p \log p)$ is increasing in p (by definition of Φ_p), and plug in a trivial upper bound $p \leq T/m$. □

Using Claim 3.5.1(c), we can write

$$\Phi_{T/|A|} = O(1) \left(\texttt{rad}(M_{\text{LP}}, \tfrac{1}{d|A|}) + \frac{M_{\text{LP}}}{B}\, \texttt{rad}(B, \tfrac{1}{d|A|})\right).$$

Rearranging the term, we obtain the bound claimed in Theorem 3.4.1.

3.6 Lower Bound

We prove that regret (1.1) obtained by algorithm PD − BwK is optimal up to poly-log factors.

Theorem 3.6.1. *Informally, any algorithm for* BwK *must incur regret*

$$\Omega\left(\min\left(\texttt{OPT},\ \texttt{OPT}\sqrt{\frac{m}{B}} + \sqrt{m\,\texttt{OPT}}\right)\right), \tag{3.29}$$

where $m = |A|$ is the number of arms and $B = \min_i B_i$ is the smallest budget.

More precisely, fix any $m \geq 2$, $d \geq 1$, $\texttt{OPT} \geq m$, and $(B_1, \ldots, B_d) \in [2, \infty)$. Let $\mathcal{F}$ be the family of all BwK *problem instances with m arms, d resources, budgets $(B_1, \ldots, B_d)$ and optimal reward* OPT*. Then any algorithm for* BwK *must incur regret (3.29) in the worst case over $\mathcal{F}$.*

We treat the two summands in Equation (3.29) separately:

Claim 3.6.2. *Consider the family $\mathcal{F}$ from Theorem 3.6.1, and let* ALG *be some algorithm for* BwK*.*

(a) ALG *incurs regret $\Omega\left(\min\left(\texttt{OPT},\ \sqrt{m\,\texttt{OPT}}\right)\right)$ in the worst case over $\mathcal{F}$.*

(b) ALG *incurs regret $\Omega\left(\min\left(\texttt{OPT},\ \texttt{OPT}\sqrt{\frac{m}{B}}\right)\right)$ in the worst case over $\mathcal{F}$.*

Theorem 3.6.1 follows from Claim 3.6.2(ab). For part (a), we use a standard lower-bounding example for MAB. For part (b), we construct a new example, specific to BwK, and analyze it using KL-divergence.

Proof of Claim 3.6.2(a). Fix $m \geq 2$ and $\texttt{OPT} \geq m$. Let $\mathcal{F}_0$ be the family of all MAB problem instances with m arms and time horizon $T = \lfloor 2\,\texttt{OPT} \rfloor$, where the "best

arm" has expected reward $\mu^* = T/\texttt{OPT}$ and all other arms have reward $\mu^* - \epsilon$ with $\epsilon = \frac{1}{4}\sqrt{m/T}$. Note that $\mu^* \in [\frac{1}{2}, \frac{3}{4}]$ and $\epsilon \leq \frac{1}{4}$. It is well-known [5] that any MAB algorithm incurs regret $\Omega(\sqrt{m\,\texttt{OPT}})$ in the worst case over $\mathcal{F}_0$.

To ensure that $\mathcal{F}_0 \subset \mathcal{F}$, let us treat each MAB instance in $\mathcal{F}_0$ as a BwK instance with d resources, budgets $(B_1, \ldots, B_d)$, and no resource consumption. □

3.6.1 The new lower-bounding example: proof of Claim 3.6.2(b)

Our lower-bounding example is very simple. There are m arms. Each arm gives reward 1 deterministically. There is a single resource with budget B.[4] The resource consumption, for each arm and each round, is either 0 or 1. The expected resource consumption is $p - \epsilon$ for the "best arm" and p for all other arms, where $0 < \epsilon < p < 1$. There is time horizon $T < \infty$. Let $\mathcal{F}_{(p,\epsilon)}$ denote the family of all such problem instances, for fixed parameters (p, ϵ). We analyze this family in the rest of this section.

Infinite time horizon. It is convenient to consider the family of problem instances which is the same as $\mathcal{F}_{(p,\epsilon)}$ except that it has the *infinite* time horizon; denote it $\mathcal{F}^{\infty}_{(p,\epsilon)}$. We will first prove the desired lower bound for this family, then extend it to $\mathcal{F}_{(p,\epsilon)}$.

The two crucial quantities that describe algorithm's performance on an instance in $\mathcal{F}^{\infty}_{(p,\epsilon)}$ is the stopping time and the total number of plays of the best arm. (Note that the total reward is equal to the stopping time minus 1.) The following claim connects these two quantities.

[4] More formally, other resources in the setting of Theorem 3.6.1 are not consumed. For simplicity, we leave them out.

Claim 3.6.3 (Stopping time)**.** *Fix an algorithm* ALG *for* BwK *and a problem instance in* $\mathcal{F}^{\infty}_{(p,\epsilon)}$*. Consider an execution of* ALG *on this problem instance. Let* τ *be the stopping time of* ALG*. For each round* t*, let* N_t *be the number of rounds* $s \leq t$ *in which the best arm is selected. Then*

$$p\,\mathbb{E}[\tau] - \epsilon\,\mathbb{E}[N_\tau] = \lfloor B + 1 \rfloor.$$

Proof. Let C_t be the total resource consumption after round t. Note that $\mathbb{E}[C_t] = pt - \epsilon N_t$. We claim that

$$\mathbb{E}[C_\tau] = \mathbb{E}[p\tau - \epsilon N_\tau]. \tag{3.30}$$

Indeed, let $Z_t = C_t - (pt - \epsilon N_t)$. It is easy to see that Z_t is a martingale with bounded increments, and moreover that $\Pr[\tau < \infty] = 1$. Therefore the Optional Stopping Theorem applies to Z_t and τ, so that $\mathbb{E}[Z_\tau] = E[Z_0] = 0$. Therefore we obtain Equation (3.30).

To complete the proof, it remains to show that $C_\tau = \lfloor B + 1 \rfloor$. Recall that ALG stops if and only if $C_t > B$. Since resource consumption in any round is either 0 or 1, it follows that $C_\tau = \lfloor B + 1 \rfloor$. □

Corollary 3.6.4. *Consider the setting in Claim 3.6.3. Then:*

(a) If ALG *always chooses the best arm then* $\mathbb{E}[\tau] = \lfloor B + 1 \rfloor / (p - \epsilon)$*.*

(b) $\texttt{OPT} = \lfloor B + 1 \rfloor / (p - \epsilon) - 1$ *for any problem instance in* $\mathcal{F}^{\infty}_{(p,\epsilon)}$*.*

(c) $p\,\mathbb{E}[\tau] - \epsilon\,\mathbb{E}[N_\tau] = (p - \epsilon)\,(1 + \texttt{OPT})$*.*

Proof. For part(b), note that we have $\mathbb{E}[\tau] \leq \lfloor B + 1 \rfloor / (p - \epsilon)$, so $\texttt{OPT} \leq \lfloor B + 1 \rfloor / (p - \epsilon) - 1$. By part (a), the equality is achieved by the policy that always selects the best arm. □

The heart of the proof is a KL-divergence argument which bounds the number of plays of the best arm. This argument is encapsulated in the following claim, whose proof is deferred to Section 3.6.2.

Lemma 3.6.5 (best arm). *Assume $p \leq \frac{1}{2}$ and $\frac{\epsilon}{p} \leq \frac{1}{16}\sqrt{\frac{m}{B}}$. Then for any* BwK *algorithm there exists a problem instance in $\mathcal{F}^{\infty}_{(p,\epsilon)}$ such that the best arm is chosen at most $\frac{3}{4}$ OPT times in expectation.*

Armed with this bound and Corollary 3.6.4(c), it is easy to lower-bound regret over $\mathcal{F}^{\infty}_{(p,\epsilon)}$.

Claim 3.6.6 (regret). *If $p \leq \frac{1}{2}$ and $\frac{\epsilon}{p} \leq \frac{1}{16}\sqrt{\frac{m}{B}}$ then any* BwK *algorithm incurs regret $\frac{\epsilon}{4p}$ OPT over $\mathcal{F}^{\infty}_{(p,\epsilon)}$.*

Proof. Fix any algorithm ALG for BwK. Consider the problem instance whose existence is guaranteed by Lemma 3.6.5. Let τ be the stopping time of ALG, and let N_t be the number of rounds $s \leq t$ in which the best arm is selected. By Lemma 3.6.5 we have $\mathbb{E}[N_\tau] \leq \frac{3}{4}\,\texttt{OPT}$. Plugging this into Corollary 3.6.4(c) and rearranging the terms, we obtain $\mathbb{E}[\tau] \leq (1+\texttt{OPT})(1-\frac{\epsilon}{4p})$. Therefore, regret of ALG is $\texttt{OPT} - (\mathbb{E}[\tau]-1) \geq \frac{\epsilon}{4p}\,\texttt{OPT}$. □

Thus, we have proved the lower bound for the infinite time horizon.

Finite time horizon. Let us "translate" a regret bound for $\mathcal{F}^{\infty}_{(p,\epsilon)}$ into a regret bound for $\mathcal{F}_{(p,\epsilon)}$.

We will need a more nuanced notation for OPT. Consider the family of problem instances in $\mathcal{F}_{(p,\epsilon)} \cup \mathcal{F}^{\infty}_{(p,\epsilon)}$ with a particular time horizon $T \leq \infty$. Let $\texttt{OPT}_{(p,\epsilon,T)}$ be the optimal expected total reward for this family (by symmetry, this quantity

does not depend on which arm is the best arm). We will write $\texttt{OPT}_T = \texttt{OPT}_{(p,\epsilon,T)}$ when parameters (p, ϵ) are clear from the context.

Claim 3.6.7. *For any fixed (p, ϵ) and any $T > \texttt{OPT}_\infty$ it holds that $\texttt{OPT}_T \geq \texttt{OPT}_\infty - \texttt{OPT}_\infty^2/T$.*

Proof. Let τ^* be the stopping time of a policy that always plays the best arm on a problem instance in $\mathcal{F}^\infty_{(p,\epsilon)}$.

$$\begin{aligned}
\texttt{OPT}_\infty - \texttt{OPT}_T &= \mathbb{E}[\tau^*] - \mathbb{E}[\min(\tau^*, T)] \\
&= \textstyle\sum_{t>T} (t - T)\ \Pr[\tau^* = t] \\
&= \textstyle\sum_{t>T}\ \Pr[\tau^* \geq t] \\
&\leq \mathbb{E}[\tau^*]/T^2 = \texttt{OPT}_\infty^2/T.
\end{aligned}$$

The inequality is due to Fact 2.4.1. □

Claim 3.6.8. *Fix (p, ϵ) and fix algorithm $\texttt{ALG}$. Let $\texttt{REG}_T$ be the regret of $\texttt{ALG}$ over the problem instances in $\mathcal{F}_{(p,\epsilon)} \cup \mathcal{F}^\infty_{(p,\epsilon)}$ with a given time horizon $T \leq \infty$. Then $\texttt{REG}_T \geq \texttt{REG}_\infty - \texttt{OPT}_\infty^2/T$.*

Proof. For each problem instance $\mathcal{I} \in \mathcal{F}^\infty_{(p,\epsilon)}$, let $\texttt{REW}_T(\mathcal{I})$ be the expected total reward of $\texttt{ALG}$ on $\mathcal{I}$, if the time horizon is $T \leq \infty$. Clearly, $\texttt{REW}_\infty(\mathcal{I}) \geq \texttt{REW}_T(\mathcal{I})$. Therefore, using Claim 3.6.7, we have:

$$\begin{aligned}
\texttt{REG}_T &= \texttt{OPT}_T - \inf_{\mathcal{I}} \texttt{REW}_T(\mathcal{I}) \\
&\geq \texttt{OPT}_T - \inf_{\mathcal{I}}\ \texttt{REW}_\infty(\mathcal{I}) \\
&= \texttt{REG}_\infty + \texttt{OPT}_T - \texttt{OPT}_\infty \\
&\geq \texttt{REG}_\infty - \texttt{OPT}_\infty^2/T.
\end{aligned}$$

□

Lemma 3.6.9 (regret: finite time horizon). *Fix $p \leq \frac{1}{2}$ and $\epsilon = \frac{p}{16}\sqrt{\frac{\min(m,B)}{B}}$. Then for any time horizon $T > 2\,\texttt{OPT}_\infty$ and any* BwK *algorithm* ALG *there exists a problem instance in $\mathcal{F}_{(p,\epsilon)}$ with time horizon T for which* ALG *incurs regret $\Omega(\texttt{OPT}_T)\sqrt{\frac{\min(m,B)}{B}}$.*

Proof. By Claim 3.6.6, ALG incurs regret $\Omega(\frac{\epsilon}{p}\,\texttt{OPT}_\infty)$ for some problem instance in $\mathcal{F}^{\infty}_{(p,\epsilon)}$. By Claim 3.6.8, ALG incurs regret $\Omega(\frac{\epsilon}{p}\,\texttt{OPT}_\infty)$ for the same problem instance in $\mathcal{F}_{(p,\epsilon)}$ with time horizon T. Finally, it is clear that $\texttt{OPT}_\infty \geq \texttt{OPT}_T$. □

Let us complete the proof of Claim 3.6.2(b). Recall that Claim 3.6.2(b) specifies the values for $(m, B, \texttt{OPT})$ that our problem instance must have. Since we have already proved Claim 3.6.2(a) and $\texttt{OPT}\sqrt{\frac{m}{B}} \leq O(\sqrt{m\,\texttt{OPT}})$ for $\texttt{OPT} < 3B$, it suffices to assume $\texttt{OPT} \geq 3B$.

Let us set ϵ as prescribed by Lemma 3.6.9. Then we obtain the desired regret for any parameter $p \leq \frac{1}{2}$ and any time horizon $T > 2\,\texttt{OPT}_{(p,\epsilon,\infty)}$. Recall that

$$\texttt{OPT}_{(p,\epsilon,\infty)} = \tfrac{\Gamma}{p}, \qquad \text{where } \Gamma = \lfloor B+1 \rfloor / \left(1 - \tfrac{1}{16}\sqrt{\tfrac{\min(m,B)}{B}}\right).$$

Thus, it remains to pick such p and T so that $\texttt{OPT}_{(p,\epsilon,T)} = \texttt{OPT}$.

Let $f(p,T) = \texttt{OPT}_{(p,\epsilon,T)}$. By Claim 3.6.7, for any $p \leq \frac{1}{2}$ and $T > 2\,\Gamma/p$ we have

$$\tfrac{\Gamma}{p} \geq f(p,T) \geq \tfrac{\Gamma}{2p}. \tag{3.31}$$

Since $\texttt{OPT} \geq 3B$, there exists $p_0 \leq \frac{1}{2}$ such that $\Gamma/p_0 \leq \texttt{OPT} \leq 2\,\Gamma/p_0$. Let $T = 8\,\Gamma/p_0$. Then Equation (3.31) holds for all $p \in [p_0/4, \frac{1}{2}]$. In particular,

$$f(p_0,T) \leq \Gamma/p_0 \leq \texttt{OPT} \leq 2\,\Gamma/p_0 \leq f(p_0/4,T).$$

Since $f(p,T)$ is continuous in p, there exists $p \in [p_0/4, \frac{1}{2}]$ such that $f(p,T) = \texttt{OPT}$.

This completes the proof of Claim 3.6.2(b) and Theorem 3.6.1.

3.6.2 The KL-divergence argument: proof of Lemma 3.6.5

Fix some BwK algorithm ALG and fix parameters (p, ϵ). Let $\mathcal{I}_a$ be the problem instance in $\mathcal{F}^{\infty}_{(p,\epsilon)}$ in which the best arm is a. For the analysis, we also consider an instance $\mathcal{I}_0$ which coincides with $\mathcal{I}_a$ but has no best arm: that is, all arms have expected resource consumption p. Let $\tau(\mathcal{I})$ be the stopping time of ALG for a given problem instance $\mathcal{I}$, and let $N_a(\mathcal{I})$ be the expected number of times a given arm a is chosen by ALG on this problem instance.

Consider problem instance $\mathcal{I}_0$. Since all arms are the same, we can apply Corollary 3.6.4(a) (suitably modified to the non-best arm) and obtain $\mathbb{E}[\tau(\mathcal{I}_0)] = \lfloor B+1 \rfloor / p$. We focus on an arm a with the smallest $N_a(\mathcal{I}_0)$. For this arm it holds that

$$N_a(\mathcal{I}_0) \leq \frac{1}{m} \sum_{a \in A} N_a(\mathcal{I}_0) = \frac{1}{m} \mathbb{E}[\tau(\mathcal{I}_0)] \leq \frac{\lfloor B+1 \rfloor}{p\, m}. \tag{3.32}$$

In what follows, we use this inequality to upper-bound $N_a(\mathcal{I}_a)$. Informally, if arm a is not played sufficiently often in $\mathcal{I}_0$, ALG cannot tell apart $\mathcal{I}_0$ and $\mathcal{I}_a$.

The *transcript* of ALG on a given problem instance $\mathcal{I}$ is a sequence of pairs $\{(a_t, c_t)\}_{t \in \mathbb{N}}$, where for each round $t \leq \tau(\mathcal{I})$ it holds that a_t is the arm chosen by ALG in round t, and c_t is the realized resource consumption in this round. For all $t > \tau(\mathcal{I})$, we define $(a_t, c_t) = (\texttt{null}, 0)$. To map this to the setup in Section 2.3, denote $\Omega = (A \cup \{\texttt{null}\}) \times \{0, 1\}$. Then the set of all possible transcripts is a subset of Ω^{∞}.

Every given problem instance $\mathcal{I}$ induces a distribution over Ω^{∞}. Let μ, ν be the distributions over Ω^{∞} that are induced by $\mathcal{I}_0$ and $\mathcal{I}_a$, respectively. We will

use the following shorthand:

$$\texttt{diff}[T_0, T_*] \triangleq \sum_{t=T_0}^{T_*} \nu(a_t = a) - \mu(a_t = a), \qquad \text{where } 1 \le T_0 \le T_* \le \infty.$$

For any $T \in \mathbb{N}$ (which we will fix later), we can write

$$N_a(\mathcal{I}_a) - N_a(\mathcal{I}_0) = \texttt{diff}[1, \infty] = \texttt{diff}[1, T] + \texttt{diff}[T+1, \infty]. \tag{3.33}$$

We will bound $\texttt{diff}[1, T]$ and $\texttt{diff}[T+1, \infty]$ separately.

Upper bound on $\texttt{diff}[1, T]$. This is where we use KL-divergence. Namely, by Equation (2.3) we have

$$\texttt{diff}[1, T] \le \tfrac{T}{2} \, \|\mu_T - \nu_T\|_1 \le T \sqrt{\tfrac{1}{2} \, \texttt{KL}(\mu_T \,\|\, \nu_T)}. \tag{3.34}$$

Now, by the chain rule (Equation (2.4)), we can focus on upper-bounding the conditional KL-divergence $\texttt{KL}_t(\mu \,\|\, \nu)$ at each round $t \le T$.

Claim 3.6.10. *For each round $t \le T$ it holds that*

$$\texttt{KL}_t(\mu \,\|\, \nu) = \mu(a_t = a) \, \left(p \, \log(\tfrac{p}{p-\epsilon}) + (1-p) \, \log(\tfrac{1-p}{1-p+\epsilon}) \right). \tag{3.35}$$

Proof. The main difficulty here is to carefully "unwrap" the definition of $\texttt{KL}_t(\mu \,\|\, \nu)$.

Fix $t \le T$ and let $\vec{w}_t \in \Omega^t$ be the partial transcript up to and including round t. For each arm y, let $f(y|\vec{w}_t)$ be the probability that ALG chooses arm y in round t, given the partial transcript $\vec{w}_t$. Let $c(y|\mathcal{I})$ be the expected resource consumption for arm y under a problem instance $\mathcal{I}$. The transcript for round $t+1$ is a pair $w_{t+1} = (a_{t+1}, c_{t+1})$, where a_{t+1} is the arm chosen by ALG in round $t+1$, and $c_{t+1} \in \{0, 1\}$ is the resource consumption in that round. Therefore if $c_{t+1} = 1$ then

$$\begin{aligned}
\mu(w_{t+1} \,|\vec{w}_t) &= f(a_{t+1}|\vec{w}_t) \; c(a_{t+1}|\mathcal{I}_0) = f(a_{t+1}|\vec{w}_t) \; p, \\
\nu(w_{t+1} \,|\vec{w}_t) &= f(a_{t+1}|\vec{w}_t) \; c(a_{t+1}|\mathcal{I}_a) = f(a_{t+1}|\vec{w}_t) \; \left(p - \epsilon \, \mathbf{1}_{\{a_{t+1}=a\}}\right).
\end{aligned}$$

Similarly, if $c_{t+1} = 0$ then

$$\mu(w_{t+1}\,|\vec{w}_t) = f(a_{t+1}|\vec{w}_t)\,(1 - c(a_{t+1}|\mathcal{I}_0)) = f(a_{t+1}|\vec{w}_t)\,(1-p),$$
$$\nu(w_{t+1}\,|\vec{w}_t) = f(a_{t+1}|\vec{w}_t)\,(1 - c(a_{t+1}|\mathcal{I}_a)) = f(a_{t+1}|\vec{w}_t)\;\left(1 - p + \epsilon\,\mathbf{1}_{\{a_{t+1}=a\}}\right).$$

It follows that

$$\log\frac{\mu(w_{t+1}\,|\vec{w}_t)}{\nu(w_{t+1}\,|\vec{w}_t)} = \mathbf{1}_{\{a_t=a\}}\;\left(\log(\tfrac{p}{p-\epsilon})\;\mathbf{1}_{\{c_{t+1}=1\}} + \log(\tfrac{1-p}{1-p+\epsilon})\;\mathbf{1}_{\{c_{t+1}=0\}}\right).$$

Taking expectations over $w_{t+1} = (a_t, c_t) \sim \mu(\cdot\,|\vec{w}_t)$, we obtain

$$\mathtt{KL}(\mu(\cdot\,|\vec{w}_t)\,\|\,\nu(\cdot\,|\vec{w}_t)) = f(x|\vec{w}_t)\;\left(p\;\log(\tfrac{p}{p-\epsilon}) + (1-p)\;\log(\tfrac{1-p}{1-p+\epsilon})\right).$$

Taking expectations over $\vec{w}_t \sim \mu_t$, we obtain the conditional KL-divergence $\mathtt{KL}_t(\mu\,\|\,\nu)$. Equation (3.35) follows because

$$\mathop{\mathbb{E}}_{\vec{w}_t\sim\mu_t} f(x|\vec{w}_t) = \mu(a_t = a). \qquad \square$$

Now we can put everything together and derive an upper bound on $\mathtt{diff}[1,T]$.

Claim 3.6.11. *Assume* $\frac{\epsilon}{p} \le \frac{1}{2}$ *and* $p \le \frac{1}{2}$. *Then* $\mathtt{diff}[1,T] \le T\,\frac{\epsilon}{p}\,\sqrt{\frac{B+1}{m}}$.

Proof. By Claim 3.6.10 and Fact 2.4.2, for each round $t \le T$ we have

$$\mathtt{KL}_t(\mu\,\|\,\nu) \le \frac{2\epsilon^2}{p}\;\mu(a_t = a).$$

By the chain rule (Equation (2.4)), we have

$$\mathtt{KL}(\mu_T\,\|\,\nu_T) \le \frac{2\epsilon^2}{p}\;\sum_{t=1}^{T}\mu(a_t = a) \le \frac{2\epsilon^2}{p}\;N_a(\mathcal{I}_0) \le 2\;\frac{B+1}{m}\;\left(\frac{\epsilon}{p}\right)^2.$$

The last inequality is the place where we use our choice of a, as expressed by Equation (3.32).

Plugging this back into Equation (3.34), we obtain $\mathtt{diff}[1,T] \le T\,\frac{\epsilon}{p}\,\sqrt{\frac{B+1}{m}}$. $\square$

Upper bound on $\texttt{diff}[T,\infty]$. Consider the problem instance $\mathcal{I}_a$, and consider the policy that always chooses the best arm. Let ν^* be the corresponding distribution over transcripts Ω^∞, and let τ be the corresponding stopping time. Note that $\nu^*(a_t = a)$ if and only if $\tau > t$. Therefore:

$$\texttt{diff}[T,\infty] \leq \sum_{t=T}^{\infty} \nu(a_t = a) \leq \sum_{t=T}^{\infty} \nu^*(a_t = a) = \sum_{t=T}^{\infty} \nu^*(\tau > t) \leq \texttt{OPT}^2/T.$$

The second inequality can be proved using a simple "coupling argument". The last inequality follows from Fact 2.4.1, observing that $\mathbb{E}[\tau] = \texttt{OPT}$.

Putting the pieces together. Assume $p \leq \frac{1}{2}$ and $\frac{\epsilon}{p} \leq \frac{1}{2}$. Denote $\gamma = \frac{\epsilon}{p}\sqrt{\frac{B+1}{m}}$. Using the upper bounds on $\texttt{diff}[1,T]$ and $\texttt{diff}[T+1,\infty]$ and plugging them into Equation (3.33), we obtain

$$N_a(\mathcal{I}_a) - N_a(\mathcal{I}_0) \leq \gamma T + \texttt{OPT}^2/T \leq \texttt{OPT}\sqrt{\gamma}$$

for $T = \texttt{OPT}/\sqrt{\gamma}$. Recall that $N_a(\mathcal{I}_0) < \texttt{OPT}/m$. Thus, we obtain

$$N_a(\mathcal{I}_a) \leq (\tfrac{1}{m} + \sqrt{\gamma})\,\texttt{OPT}.$$

Recall that we need to conclude that $N_a(\mathcal{I}_a) \leq \frac{3}{4}\texttt{OPT}$. For that, it suffices to have $\gamma \leq \frac{1}{16}$.

3.7 Applications

Owing to its generality, the BwK problem admits applications in many different domains. We describe some of these applications below. Instantiating specific regret bounds is straightforward for most of these applications, as soon as one precisely defines the setting.

BwK with discretization. In many applications of BwK the action space A is very large or infinite, so the algorithms developed in the previous sections are not immediately applicable. However, in these applications the action space has some structure that our algorithms can leverage. For example, in dynamic pricing (with one type of good) every possible action (arm) corresponds to a price – i.e., a number in some fixed interval.

To handle such applications, we use a simple approach called *uniform discretization*: we select a subset $S \subset A$ of arms which are, in some sense, uniformly spaced in A, and apply a BwK algorithm for this subset. In the dynamic pricing application, S can consist of all prices in the allowed interval that are of the form $\ell\epsilon$, $\ell \in \mathbb{N}$, for some fixed discretization parameter $\epsilon > 0$. We call this subset the *additive ϵ-mesh*. The granularity of discretization (i.e., the value of ϵ) is adjusted in advance so as to minimize regret.

This generic approach has been successfully used in past work on MAB on metric spaces (e.g., [34, 32, 36, 42]) and dynamic pricing (e.g., [35, 15, 13, 7]) to provide worst-case optimal regret bounds. To carry it through to the setting of BwK, we need to define an appropriate notion of discretization (which would generalize the additive ϵ-mesh for dynamic pricing), and argue that it does not cause too much damage. Compared to the usage of discretization in prior work, we need to deal with two technicalities. First, we need to argue about distributions over arms rather than individual arms. Second, in order to compare an arm with its "image" in the discretization, we need to consider the difference in the ratio of expected reward to expected consumption, rather than the difference in rewards.[5] We flesh out the details in Section 3.8.

[5]Because of these technicalities, our approach to discretization does not immediately extend to some of the generalizations of dynamic pricing and dynamic procurement that we consider in the subsequent subsections.

For dynamic pricing, we show that if we use the additive ϵ-mesh S_ϵ, and $R(S_\epsilon)$ is the regret on the problem instance restricted to S_ϵ, then regret on the original problem instance is at most $\epsilon\, d\, B + R(S_\epsilon)$, where d is the number of constraints. More generally, we prove this for any BwK domain, and any subset $S_\epsilon \subset A$ which satisfies some axioms for a given parameter ϵ; we call such S_ϵ an *ϵ-discretization*. Once we define an ϵ-discretization S_ϵ for every $\epsilon > 0$, then in order to optimize the regret bound $\epsilon\, d\, B + R(S_\epsilon)$ up to constant factors it suffices to pick ϵ such that $\epsilon\, dB = R(S_\epsilon)$. We need to consider regret bounds $R(S_\epsilon)$ that are in terms of the maximal possible expected total reward M_{LP} rather than the (possibly smaller) $\texttt{OPT}$, since the value of $\texttt{OPT}$ is not initially known to the algorithm.[6]

3.7.1 Dynamic pricing with limited supply

Basic version: unit demands. In the basic version of the *dynamic pricing problem*, the algorithm is a seller which has k identical items for sale and is facing n agents (potential buyers) that are arriving sequentially. Each agent is interested in buying one item. The algorithm offers a take-it-or-leave-it price p_t to each arriving agent t. The agent has a fixed value $v_t \in [0, 1]$ for an item, which is *private*: not known to the algorithm. The agent buys an item if and only if $p_t \geq v_t$. We assume that each v_t is an independent sample from some fixed (but unknown) distribution, called the buyers' *demand distribution*. The goal of the algorithm is to maximize his expected revenue.

[6]For problem instances with $\texttt{OPT} \ll M_{\text{LP}}$, additional information about the problem instance may help the algorithm: it may effectively reduce the problem to a smaller BwK domain with a smaller value of M_{LP}, which would allow the algorithm to choose ϵ more efficiently so as to reduce the number of arms in the ϵ-discretization and hence reduce the resulting regret.

This problem has been studied in [35, 15, 13, 7]. The best result is an algorithm which achieves regret $\tilde{O}(k^{2/3})$ compared to the best fixed price [7];[7] this regret rate is proved to be worst-case optimal. For demand distributions that satisfy a standard (but limiting) assumption of *regularity*, the best fixed price is essentially as good as the best dynamic policy.

Dynamic pricing is naturally interpreted as a BwK domain: arms correspond to prices, rounds correspond to arriving agents (so there is a time horizon of n), and there is a single type of resource: the k items. One can think of the agents as simply units of time, so that n is the time horizon. Let $S(p)$ be the probability of a sale at price p. The latent structure μ is determined by the function $S(\cdot)$: the expected reward is $r(p,\mu) = p\,S(p)$, and expected resource consumption is simply $c(x,\mu) = S(p)$. Since there are only k items for sale, it follows that $\texttt{OPT} \leq k$. (To analyze `Mixture_Elimintation`, note that the LP-values are bounded above by k, so we can take $M_{\text{LP}} = k$.) Prices can be discretized using the ϵ-uniform mesh over $[0,1]$, resulting in $\frac{1}{\epsilon}$ arms.

We recover the optimal $\tilde{O}(k^{2/3})$ regret result from [7] with respect to offering the same price to each buyer, and moreover extend this result to regret with respect to the optimal dynamic policy. We observe that the expected revenue of the optimal dynamic policy can be twice the expected revenue of the best fixed price (see Section 3.9). The crucial technical advance compared to the prior work is that our algorithms strive to converge to an optimal *distribution* over prices, whereas the algorithms in prior work target the best fixed price. Technically, the result follows by applying Theorem 3.8.2 for the uniform ϵ-mesh, in conjunction with either of the two main algorithms, and optimizing the ϵ.

[7] The $\tilde{O}(\cdot)$ notation hides poly-log factors.

Extension: non-unit demands. Agents may be interested in buying more than one unit of the product. Accordingly, let us consider an extension where an algorithm can offer each agent multiple units. More specifically: in each round t, the algorithm offers up to λ_t units at a fixed price p_t per unit, where the pair (p_t, λ_t) is chosen by the algorithm. The t-th agent then chooses how many units to buy. Agents' valuations may be non-linear in k_t, the number of units they receive. Each agent chooses k_t that maximizes her utility. We restrict $\lambda_t \leq \Lambda$, where Λ is an a-priori chosen parameter.

Let us solve this generalization via BwK. We model it as a BwK domain with a single resource and action space $A = [0,1] \times \{0, 1, \ldots, \Lambda\}$. Note that for each arm $a = (p_t, \lambda_t)$ the expected reward is $r(a, \mu) = p_t \mathbb{E}[k_t]$, and the expected consumption is $c(a, \mu) = \mathbb{E}[k_t]$. As before, we can use the additive ϵ-mesh on $[0,1]$ for discretization; this results in Λ/ϵ arms. Optimizing the ϵ, we obtain regret $\tilde{O}(k\,\Lambda)^{2/3}$. [8]

It is also interesting to consider a more restricted version where $\lambda_t = \Lambda$. Then in each round t the algorithm only chooses the price p_t, so that the action space is A. A similar argument gives regret $\tilde{O}(k^{2/3}\Lambda^{1/3})$.

Other extensions. The generality of BwK allows to handle several other extensions. For these extensions, it appears more difficult to bound the "damage" due to discretization. We will assume that the action space is restricted to a specific finite subset $S \subset A$ that is given to the algorithm, such as the additive ϵ-mesh for some specific ϵ. We bound the *S-regret*: regret with respect to the value of $\texttt{OPT}_{\text{LP}}$

[8] Formally, the definition of BwK requires that the per-round reward and the per-round resource consumption are at most 1. Therefore we scale down the rewards and the consumption by the factor of Λ; effectively, the supply constraint is divided by Λ, and regret is multiplied by Λ. After the re-scaling, the LP-values are upper-bounded by k, so that we can take $M_{\text{LP}} = k$.

on the restricted action space. Such regret bounds depend on $|S|$.

- *Multiple products.* The algorithm has d products for sale, with k_i units of each product i. (To simplify regret bounds, let us assume $k_i = k$.) In each round t, an agent arrives. This agent is interested in buying only one unit of each product. The agent is characterized by the vector of values $(v_{t,1}, \ldots, v_{t,d}) \in [0,1]^d$, one for each product i. This vector is private: not known to the algorithm. We assume that it is an independent sample from a fixed but unknown distribution over $[0,1]^d$; note that arbitrary correlations between values of different products are allowed. The algorithm offers a vector of prices $(p_{t,1}, \ldots, p_{t,d}) \in [0,1]^d$, one for each product i. The agent buys one unit of each product i such that $p_{t,i} \geq v_{t,i}$.

This problem corresponds to a BwK domain with d resources, one for each product. Arms are price vectors, so that $A = [0,1]^d$. We assume that the algorithm is given a restricted action space $S \subset A$, such as an additive ϵ-mesh for some specific ϵ. We obtain S-regret $\tilde{O}(d\sqrt{k\,|S|})$. [9]

Similarly, we can handle a version where the t-th agent buys only one item; that is, she buys one unit of type $i^* = \text{argmax}_i\, v_i - p_i$, if and only if $v_{i^*} \geq p_{i^*}$. We obtain S-regret $\tilde{O}(\sqrt{d\,k\,|S|})$.

- *Network revenue management.* More generally, an algorithm may have d products for sale which may be produced on demand from limited *primitive resources*, so that each unit of each product i consumes a fixed and known amount $c_{ij} \in [0,1]$ of each primitive resource j. This generalization is known as network revenue management problem (see [14] and references therein). All other de-

[9] To ensure that the maximal per-round reward is at most 1, as required by BwK, we re-scale all rewards down by the factor of d; effectively, regret is increased by the factor of d. After the re-scaling, $\texttt{OPT} \leq M_{\texttt{LP}} \leq k$.

tails are the same as above; for simplicity, let us focus on a version in which each agent buys at most one item.

This corresponds to a BwK domain with the same action space as above: $A = [0,1]^d$. For any given restricted action space $S \subset A$ with m arms, we obtain S-regret given by Equation (1.1). In particular, if all resource constraints (including the time horizon) are scaled up by factor γ, regret scales as $\sqrt{\gamma}$. This improves over the main result in [14], where (essentially) regret is stated in terms of γ and scales as $\gamma^{2/3}$.

- *Bundling and volume pricing.* When selling to agents with non-unit demands, an algorithm may use discounts and/or surcharges for buying multiple units of a product (the latter may make sense for high-valued products such as tickets to events at the Olympics). More generally, an algorithm can may use discounts and/or surcharges for some *bundles* of products, where each bundle can include multiple units of multiple products (e.g.: two beers and one snack). In full generality, there is a collection $\mathcal{F}$ of allowed bundles. In each round an algorithm offers a *menu* of options which consists of a price for every allowed bundle in $\mathcal{F}$. Each buyer has a private valuation for every bundle; for each product, she chooses the bundle which maximizes her utility. The vector of private valuations over bundles comes from a fixed but unknown distribution.

This corresponds to a BwK domain where arms correspond to the feasible menus. In other words, each arm is a price vector over all possible bundles in $\mathcal{F}$, so that the (full) action space is $A = [0,1]^{\mathcal{F}}$. We assume that the algorithm is given a restricted action space $S \subset A$. If each feasible bundle contains at most ℓ units, we obtain S-regret $\tilde{O}(\sqrt{d\,\ell\,k\,|S|})$.

We can reduce the "dimensionality" of the action space by restricting the bundle pricing. For example, the algorithm can offer a volume discounts of $x\%$ compared to the price of a single item, where x depends only on the number of items in the bundle, but not on the specific products in this bundle and not on the prices of a single unit of these products.

- *Buyer targeting.* Suppose there are ℓ different types of buyers (say *men* and *women*), and the demand distribution of a buyer depends on her type. The buyer type is modeled as a sample from a fixed but unknown distribution. In each round the seller observes the type of the current buyer (e.g., using a *cookie* or a user profile), and can choose the price depending on this type.

This can be modeled as a BwK domain where arms correspond to functions from buyer types to prices. For example, with ℓ buyer types and a single product, the (full) action space is $A = [0,1]^\ell$. Assuming we are given a restricted action space $S \subset A$, we obtain S-regret $\tilde{O}(\sqrt{k\,|S|})$.

Our regret guarantees for the above extensions are with respect to the optimal dynamic policy on the restricted action space. It is worth emphasizing that the benchmark equivalence result (the best fixed action is almost as good as the best dynamic policy, for regular demand distributions) applies only to the basic version of dynamic pricing. For example, no such result is known for multiple types of goods, even if the demand distribution for each individual type is regular. (See Section 3.9 for a related discussion.)

3.7.2 Dynamic procurement and crowdsourcing

Basic version: unit supply. A "dual" problem to dynamic pricing is *dynamic procurement*, where the algorithm is buying rather than selling. The algorithm has a budget B to spend, and is facing n agents (potential sellers) that are arriving sequentially. Each seller is interested in selling one item. Each seller's value for an item is an independent sample from some fixed (but unknown) distribution with support $[0, 1]$. The algorithm offers a take-it-or-leave-it price to each arriving agent. The goal is to maximize the number of items bought. This problem has been studied in [9]; they achieve a multiplicative constant-factor approximation with respect to the optimal dynamic policy.

Application to crowdsourcing. The problem is particularly relevant to the emerging domain of *crowdsourcing*, where agents correspond to the (relatively inexpensive) workers on a crowdsourcing platform such as Amazon Mechanical Turk, and "items" bought/sold correspond to simple jobs ("microtasks") that can be performed by these workers. The algorithm corresponds to the "requester": an entity that submits jobs and benefits from them being completed. The (basic) dynamic procurement model captures an important issue in crowdsourcing that a requester interacts with multiple users with unknown values-per-item, and can adjust its behavior (such as the posted price) over time as it learns the distribution of users. While this basic model ignores some realistic features of crowdsourcing environments, some of these limitations are addressed by the generalizations which we present below.

Dynamic procurement via `BwK`. Let us cast dynamic procurement as a BwK domain. As in dynamic pricing, agents correspond to time rounds, and n is the

time horizon. The only resource is money. Arms correspond to prices: $A = [0, 1]$. Letting $S(p)$ be the probability of a sale at price p, the expected reward (items bought) is $r(p, \mu) = S(p)$, and the expected resource consumption is $c(p, \mu) = pS(p)$. The best a-priori upper bound on the LP-value is $M_{\text{LP}} = n$.

We find that arbitrarily small prices are not amenable to discretization. Instead, we focus on prices $p \geq p_0$, where $p_0 \in (0, 1)$ is a parameter to be adjusted, and construct an ϵ-discretization on the set $[p_0, 1]$. We find that the additive δ-mesh (for a suitably chosen δ) is not the most efficient way to construct an ϵ-discretization in this domain. Instead, we use a mesh of the form $\{\frac{1}{1+j\epsilon} : j \in \mathbb{N}\}$ (we call it the *hyperbolic ϵ-mesh*). Then we obtain an ϵ-discretization with significantly fewer arms. Optimizing the parameters p_0 and ϵ, we obtain regret $\tilde{O}\left(n/B^{1/4}\right)$; see Section 3.8.1 for more details.

Our result is meaningful, and improves over the constant-factor approximation in [9], as long as OPT is not too small compared to $n/B^{1/4}$. Recall that our result is with respect to the optimal dynamic policy. We observe that in this domain the optimal dynamic policy can be vastly superior compared to the best fixed price (see Section 3.9 for a specific example).

Extension: non-unit supply. As in dynamic pricing, we consider an extension where each agent may be interested in more than one item. For example, a worker may be interested in performing several jobs. In each round t, the algorithm offers to buy up to λ_t units at a fixed price p_t per unit, where the pair (p_t, λ_t) is chosen by the algorithm. The t-th agent then chooses how many units to sell. Agents' valuations may be non-linear in k_t, the number of units they sell. Each agent chooses k_t that maximizes her utility. We restrict $\lambda_t \leq \Lambda$, where Λ is an a-priori chosen parameter.

We model it as a BwK domain with a single resource (money) and action space $A = [0,1] \times \{0, 1, \ldots, \Lambda\}$. Note that for each arm $a = (p_t, \lambda_t)$ the expected reward is $r(a, \mu) = \mathbb{E}[k_t]$, and the expected consumption is $c(a, \mu) = p_t \, \mathbb{E}[k_t]$. As in the basic dynamic procurement problem, we focus on prices $p \geq p_0$ and use the hyperbolic ϵ-mesh on $[p_0, 1]$ for discretization, for some parameters $p_0, \epsilon \in (0, 1)$. Optimizing the ϵ, we obtain regret $\tilde{O}(\Lambda^{3/2} n / B^{1/4})$; see Section 3.8.1 for more details.

In a more restricted version with $\lambda_t = \Lambda$, we obtain regret $\tilde{O}(\Lambda^{5/4} n / B^{1/4})$.

Other extensions. Further, we can model and handle a number of other extensions. As in "other extensions" for dynamic pricing, will assume that the action space is restricted to a specific finite subset $S \subset A$ that is given to the algorithm, and bound the *S-regret* – regret with respect to the restricted action space.

• *Multiple types of jobs*. There are ℓ types of jobs requested on the crowdsourcing platform. Each agent t has a private cost $v_{t,i} \in [0,1]$ for each type i; the vector of private costs comes from a fixed but unknown distribution (i.e., arbitrary correlations are allowed). The algorithm derives utility $u_i \in [0,1]$ from each job of type i. In each round t, the algorithm offers a vector of prices $(p_{t,1}, \ldots, p_{t,\ell})$, where $p_{t,i}$ is the price for one job of type i. For each type i, the agent performs one job of this type if and only if $p_{t,i} \geq v_{t,i}$, and receives payment $p_{t,i}$ from the algorithm.

Here arms correspond to the ℓ-dimensional vectors of prices, so that the (full) action space is $A = [0,1]^\ell$. Given the restricted action space $S \subset A$, we obtain S-regret $\tilde{O}(\ell)(\sqrt{n\,|S|} + n\sqrt{\ell\,|S|/B})$.

• *Additional features.* We can also model more complicated "menus" so that each agent can perform several jobs of the same type. Then in each round, for each type i, the algorithm specifies the maximal offered number of jobs of this type and the price per one such job.

We can also incorporate constraints on the maximal number of jobs of each type that is needed by the requester, and/or the maximal amount of money spend on each type.

• *Competitive environment.* There may be other requesters in the system, each offering its own vector of prices in each round. (This is a realistic scenario in crowdsourcing, for example.) Each seller / worker chooses the requester and the price that maximize her utility. One standard way to model such competitive environment is to assume that the "best offer" from the competitors is a vector of prices which comes from a fixed but unknown distribution. This can be modeled as a `BwK` instance with a different distribution over outcomes which reflects the combined effects of the demand distribution of agents and the "best offer" distribution of the environment.

3.7.3 Other applications to Electronic Markets

Ad allocation with unknown click probabilities. Consider *pay-per-click* (PPC) advertising on the web (in particular, this is a prevalent model in sponsored search auctions). The central premise in PPC advertising is that an advertiser derives value from her ad only when the user clicks on this ad. The ad platform allocates ads to users that arrive over time.

Consider the following simple (albeit highly idealized) model for PPC ad allocation. Users arrive over time, and the ad platform needs to allocate an ad to each arriving user. There is a set A of available ads. Each ad a is characterized by the payment-per-click π_a and click probability μ_a; the former quantity is known to the algorithm, whereas the latter is not. If an ad a is chosen, it is clicked on with probability μ_a, in which case payment π_a is received. The goal is to maximize the total payment. This setting, and various extensions thereof that incorporate user/webpage context, has received a considerable attention in the past several years (starting with [44, 45, 40]). In fact, the connection to PPC advertising has been one of the main motivations for the recent surge of interest in MAB.

We enrich the above setting by incorporating advertisers' *budgets*. In the most basic version, for each ad a there is a budget B_a — the maximal amount of money that can be spent on this ad. More generally, an advertiser can have an ad campaign which consists of a subset S of ads, so that there is a per-campaign budget S. Even more generally, an advertiser can have a more complicated budget structure: a family of overlapping subsets $S \subset A$ and a separate budget B_S for each S. For example, BestBuy can have a total budget for the ad campaign, and also separate budgets for ads about TVs and ads about computers. Finally, in addition to budgets (i.e., constraints on the number of times ads are clicked), an advertiser may wish to have similar constraints on the number of times ads are shown. `BwK` allows us to express all these constraints.

Adjusting a repeated auction. An auction is held in every round, with a fresh set of participants. The number of participants and a vector of their types come from a fixed but unknown distribution. The auction is *adjustable*: it has some

parameter that can be adjusted by the auctioneer. For example, [18] studies a repeated second price auction with adjustable reserve price. The auctioneer has a limited inventory and wishes to optimize revenue.

The prior work [18] considers the case of a single type of good and unlimited inventory. They design an algorithm with $O(\sqrt{n \log n})$ regret, where n is the number of rounds; this regret is proved optimal.

BwK allows us to model limited inventory. For example, for the setting in [18] our algorithm achieves $O(k \log n)^{2/3}$ regret, where k is the number of items. Moreover, we can incorporate any auction design with adjustable parameter that admits discretization (as defined in Section 3.8). Further, we can handle multiple types of goods; such extension would correspond to multiple resource constraints in BwK.

Repeated bidding. A bidder participates in a repeated auction, such as a sponsored search auction. In each round t, the bidder can adjust her bid b_t based on the past performance. The outcome for this bidder is a vector (p_t, u_t), where p_t is the payment and u_t is the utility received. We assume that this vector comes from a fixed but unknown distribution. The bidder has a fixed budget.

We model this as a BwK problem where arms correspond to the possible bids, and the single resource is money. Note that (the basic version of) dynamic procurement corresponds to this setting with two possible outcome vectors (p_t, u_t): $(0, 0)$ and $(b_t, 1)$.

The BwK setting also allows to incorporate more complicated constraints. For example, an action can result in several different types of outcomes that are useful for the bidder (e.g., an ad shown to a *male* or an ad shown to a *female*),

but the bidder is only interested in a limited quantity of each outcome.

3.7.4 Application to network routing and scheduling

In addition to applications to Electronic Markets, we describe two applications to network routing and scheduling. In both applications an algorithm choose between different feasible policies to handle arriving "service requests", such as connection requests in network routing and jobs in scheduling.

Adjusting a routing protocol. Consider the following stylized application to routing in a communication network. Connection requests arrive one by one. A connection request consists of a pair of terminals; assume the pair comes from a fixed but unknown distribution. The system needs to choose a *routing protocol* for each connection, out of several possible routing protocols. The routing protocol defines a path that connects the terminals; abstractly, each protocol is simply a mapping from terminal pairs to paths. Once the path is chosen, a connection between the terminals is established. Connections persist for a significant amount of time. Each connection uses some amount of bandwidth. For simplicity, we can assume that this amount is fixed over time for every connection, and comes from a fixed but unknown distribution (although even a deterministic version is interesting). Each edge in the network (or perhaps each node) has a limited capacity: the total bandwidth of all connections that pass though this edge or node cannot exceed some value. A connection which violates any capacity constraint is terminated. The goal is to satisfy a maximal number of connections.

We model this problem as `BwK` as follows: arms correspond to the feasible

routing protocols, each edge/node is a limited resource, each satisfied connection is a unit reward.

Further, if the time horizon is partitioned in epochs, we can model different bandwidth utilization in each phase; then a resource in BwK is a pair (edge,epoch).

Adjusting a scheduling policy. An application with a similar flavor arises in the domain of scheduling long-running jobs to machines. Suppose jobs arrive over time. Each job must be assigned to one of the machines (or dropped); once assigned, a job stays in the system forever (or for some number of "epochs"), and consumes some resources. Jobs have multiple "types" that can be observed by the scheduler. For each type, the resource utilization comes from a fixed but unknown distribution. Note that there may be multiple resources being consumed on each machine: for example, jobs in a datacenter can consume CPU, RAM, disk space, and network bandwidth. Each satisfied job of type i brings utility u_i. The goal of the scheduler is to maximize utility given the constrained resources.

The mapping of this setting to BwK is straightforward. The only slightly subtle point is how to define the arms: in BwK terms, arms correspond to all possible mappings from job types to machines.

One can also consider an alternative formulation where there are several allowed scheduling policies (mappings from types and current utilization of resources to machines), and in every round the scheduler can choose to use one of these policies. Then the arms in BwK correspond to the allowed policies.

3.8 BwK with discretization

In this section we flesh out our approach to BwK with uniform discretization, as discussed early in Section 3.7. The technical contribution here is a definition of the appropriate structure, and proof that discretization does not do much damage.

Definition 3.8.1. We say that arm a *ϵ-covers* arm y if the following two properties are satisfied for each resource i and every latent structure μ in a given BwK domain:

(i) $r(a,\mu)/c_i(a,\mu) \geq r(y,\mu)/c_i(y,\mu) - \epsilon$.

(ii) $c_i(a,\mu) \geq c_i(y,\mu)$.

A subset $S \subset A$ of arms is called an *ϵ-discretization* of A if each arm in A is ϵ-covered by some arm in S.

Note that we consider the difference in the ratio of expected reward to expected consumption, whereas for MAB in metric spaces it suffices to consider the difference in expected rewards. For a typical application of uniform discretization it holds that $A \subset \mathbb{R}^\ell$ for some $\ell \in \mathbb{N}$, and a suitably chosen mesh on A is an ϵ-discretization. There are several types of meshes one could consider, depending on the particular BwK domain: *additive δ-mesh* $\delta\,\mathbb{N}^\ell$, *multiplicative δ-mesh* $\{(1-\delta)^j : j \in \mathbb{N}\}^\ell$, and *hyperbolic δ-mesh* $\{\frac{1}{1+j\delta} : j \in \mathbb{N}\}^\ell$. For a particular mesh $M \subset \mathbb{R}^\ell$, the "mesh on A" is simply $A \cup M$.

Once we construct an ϵ-discretization S of A, we can run a BwK algorithm on S and obtain expected total reward competitive with $\texttt{OPT}_{\text{LP}}(S)$, the value of

$\texttt{OPT}_{\texttt{LP}}$ if the action space is restricted to S. This makes sense if $\texttt{OPT}_{\texttt{LP}}(S)$ is not much worse than $\texttt{OPT}_{\texttt{LP}}(A)$.

Theorem 3.8.2 (BwK: discretization)**.** *Fix a BwK domain with action space A. Consider an algorithm* ALG *which achieves expected total reward* $\texttt{REW} \geq \texttt{OPT}_{\texttt{LP}}(S) - R(S)$ *if applied to the restricted action space $S \subset A$, for any given $S \subset A$. If S is an ϵ-discretization of A, for some $\epsilon \geq 0$, then*

$$\texttt{REW} \geq \texttt{OPT}_{\texttt{LP}}(A) - \epsilon\, d\, B - R(S).$$

The technical content in Theorem 3.8.2 is that $\texttt{OPT}_{\texttt{LP}}(S)$ is not much worse than $\texttt{OPT}_{\texttt{LP}}(A)$. Such result is typically trivial in MAB settings where the benchmark is the best fixed arm, but requires some work in our setting because we need to consider distributions over arms.

Lemma 3.8.3. *Consider an instance of* BwK *with action space A. Let $S \subset A$ be an ϵ-discretization of A, for some $\epsilon \geq 0$. Then* $\texttt{OPT}_{\texttt{LP}}(S) \geq \texttt{OPT}_{\texttt{LP}}(A) - \epsilon\, d\, B$

Proof. Let $\mathcal{D}$ be the distribution over arms in A which maximizes $\texttt{LP}(\mathcal{D}, \mu)$. We use $\mathcal{D}$ to construct a distribution $\mathcal{D}_S$ over S which is nearly as good.

Let μ be the (actual) latent structure. For brevity, we will suppress μ from the notation: $c_i(a) = c_i(a, \mu)$ and $c_i(\mathcal{D}) = c_i(\mathcal{D}, \mu)$ for arms a, distributions $\mathcal{D}$ and resources i. Similarly, we will write $r(a) = r(a, \mu)$ and $r(\mathcal{D}) = r(\mathcal{D}, \mu)$.

We define $\mathcal{D}_S$ as follows. Since S is an ϵ-discretization of A, there exists a family of subsets $(\texttt{cov}(a) \subset A : x \in A)$ so that each arm a ϵ-covers all arms in $\texttt{cov}(a)$, the subsets are disjoint, and their union is A. Fix one such family of subsets, and define

$$\mathcal{D}_S(a) = \sum_{y \in \texttt{cov}(a)} \mathcal{D}(y) \min_i \frac{c_i(y)}{c_i(a)}, \quad x \in S.$$

Note that $\sum_{x \in S} \mathcal{D}_S(S) \leq 1$ by Definition 3.8.1(ii). With the remaining probability, the null arm is chosen (i.e., the algorithm skips a given round).

To argue that $\texttt{LP}(\mathcal{D}_S, \mu)$ is large, we upper-bound the resource consumption $c_i(\mathcal{D}_S)$, for each resource i, and lower-bound the reward $r(\mathcal{D}_S)$.

$$\begin{aligned}
c_i(\mathcal{D}_S) &= \textstyle\sum_{x \in S} \; c_i(a)\, \mathcal{D}_S(a) \\
&\leq \sum_{x \in S} c_i(a) \sum_{y \in \texttt{cov}(a)} \mathcal{D}(y) \frac{c_i(y)}{c_i(a)} = \sum_{x \in S} \sum_{y \in \texttt{cov}(a)} \mathcal{D}(y)\, c_i(y) = \sum_{y \in A} \mathcal{D}(y) c_i(y) \\
&= c_i(\mathcal{D}) \qquad (3.36)
\end{aligned}$$

(Note that the above argument did not use the property (i) in Definition 3.8.1.)

$$\begin{aligned}
r(\mathcal{D}_S) &= \textstyle\sum_{x \in S} \; r(a)\, \mathcal{D}_S(a) \\
&= \sum_{x \in S} r(a) \sum_{y \in \texttt{cov}(a)} \mathcal{D}(y) \; \min_i \; \frac{c_i(y)}{c_i(a)} \\
&= \sum_{x \in S} \sum_{y \in \texttt{cov}(a)} \mathcal{D}(y) \; \min_i \; \frac{c_i(y)\, r(a)}{c_i(a)} \\
&\geq \sum_{x \in S} \sum_{y \in \texttt{cov}(a)} \mathcal{D}(y) \; \min_i \; r(y) - \epsilon c_i(y) \qquad \text{(by Definition 3.8.1(i))} \\
&= \textstyle\sum_{y \in A} \mathcal{D}(y) \; \min_i \; r(y) - \epsilon c_i(y) \\
&\geq \textstyle\sum_{y \in A} \mathcal{D}(y) \; (r(y) - \epsilon \; \sum_i c_i(y)) \\
&= r(\mathcal{D}) - \epsilon \textstyle\sum_i c_i(\mathcal{D}). \qquad (3.37)
\end{aligned}$$

Let $\tau(\mathcal{D}) = \min_i \frac{B}{c_i(\mathcal{D})}$ be the stopping time in the linear relaxation, so that $\texttt{LP}(\mathcal{D}, \mu) = \tau(\mathcal{D})\, r(\mathcal{D})$. By Equation (3.36) we have $\tau(\mathcal{D}_S) \geq \tau(\mathcal{D})$. We are ready

for the final computation:

$$\begin{aligned}
\mathtt{LP}(\mathcal{D}_S, \mu) &= \tau(\mathcal{D}_S)\; r(\mathcal{D}_S) \\
&\geq \tau(\mathcal{D})\; r(\mathcal{D}_S) \\
&\geq \tau(\mathcal{D})\,(r(\mathcal{D}) - \epsilon \textstyle\sum_i c_i(\mathcal{D})) && \text{(by Equation (3.37))} \\
&\geq r(\mathcal{D})\,\tau(\mathcal{D}) - \epsilon\,\tau(\mathcal{D}) \textstyle\sum_i c_i(\mathcal{D}) \\
&\geq \mathtt{LP}(\mathcal{D}, \mu) - \epsilon\, d\, B. && \square
\end{aligned}$$

3.8.1 Discretization for dynamic procurement

A little more work is needed to apply uniform discretization for dynamic procurement. Consider dynamic procurement with non-unit supply, as defined in Section 3.7. Throughout this subsection, assume n agents, budget B, and ceiling Λ. Recall that in each round t the algorithm makes an offer of the form (p_t, λ_t), where $p_t \in [0,1]$ is the posted price per unit, and $\lambda_t \leq \Lambda$ is the maximal number of units offered for sale in this round. The action space here is $A = [0,1] \times \{1, \ldots, \Lambda\}$, the set of all possible (p, λ) pairs.

For each arm $a = (p, \lambda)$, let $S(a)$ be the expected number of items sold. Recall that expected consumption is $c(a) = p\,S(a)$, and expected reward is $S(a)$. It follows that $\frac{r(a)}{c(a)} = \frac{1}{p}$. By Definition 3.8.1 price p ϵ-covers price $q < p$ if and only if $\frac{1}{p} \geq \frac{1}{q} - \epsilon$.

It is easy to see that the hyperbolic ϵ-mesh S on $[0,1]$ is an ϵ-discretization: namely, each arm (q, λ) is ϵ-covered by (q, λ), where p is the smallest price in S such that $p \geq q$. Unfortunately, such S has infinitely many points. In fact, it is easy to see that any ϵ-discretization on A must be infinite (even for $\Lambda = 1$). To ob-

tain a finite ϵ-discretization, we only consider prices $p \geq p_0$, for some parameter p_0 to be tuned later. We argue that this restriction is not too damaging:

Claim 3.8.4. *Consider dynamic procurement with non-unit supply. Then for any $p_0 \in (0,1)$ it holds that*

$$\texttt{OPT}_{\texttt{LP}}(\{0\} \cup [p_0, 1]) \geq \texttt{OPT}_{\texttt{LP}} - \Lambda^2 p_0 n^2 / B.$$

Proof. When $p_0 > B/(n\Lambda)$ the bound is trivial and for the rest of the proof we assume that $p_0 \leq B/(n\Lambda)$. Let $\mathcal{D}$ be the distribution over arms which maximizes $\texttt{LP}(\mathcal{D}, \mu)$ to get $\texttt{OPT}_{\texttt{LP}}$. By Claim 3.1.2, we can further assume that $\mathcal{D}$ is LP-perfect. Thus, $\mathcal{D}$ has a support set of size at most 2, say arms $a_i = (p_i, \lambda_i)$, $i = 1, 2$, where p_i is the posted price and λ_i is the maximal allowed number of items. W.l.o.g. assume $p_1 \leq p_2$. Note that p_1 can be 0, which would correspond to the "null arm". Moreover, letting s_i denote the expected number of items bought for arm (p_i, λ_i), then by LP-perfectness the expected consumption is

$$c(\mathcal{D}, \mu) = \sum_{i=1}^{2} \mathcal{D}(a_i)\, p_i s_i \leq B/n. \tag{3.38}$$

(Formally, to apply Claim 3.1.2 one needs to divide the rewards and consumptions (and hence the budget) by Λ, so that consumptions and rewards in each round are at most 1.)

Let us define a distribution $\mathcal{D}'$ which has support in $\{0\} \cup [p_0, 1]$.

- If $p_1 \geq p_0$ then define $\mathcal{D}' = \mathcal{D}$ and we get $\texttt{OPT}_{\texttt{LP}}(\{0\} \cup [p_0, 1]) \geq \texttt{OPT}_{\texttt{LP}}$.
- From here on, assume $p_1 < p_0$. If $\mathcal{D}$ is a distribution of support 1 (i.e $\mathcal{D}'(a_2) = 0$) then define $\mathcal{D}'(B/(n\Lambda), \lambda_1) = 1$ and we get $\texttt{OPT}_{\texttt{LP}}(\{0\} \cup [p_0, 1]) \geq \texttt{OPT}_{\texttt{LP}}$.

- If $\mathcal{D}$ is a distribution of support 2 then by LP-perfectness Equation (3.38) is satisfied with equality. Therefore $p_2 \geq B/(ns_2) \geq B/(n\Lambda)$. Define

$$\begin{aligned}\mathcal{D}'(p_0, \lambda_1) &= \mathcal{D}(p_1, \lambda_1)\\ \mathcal{D}'(p_2, \lambda_2) &= \max(0, \mathcal{D}(p_2, \lambda_2) - \frac{p_0\Lambda}{p_2})\\ \mathcal{D}'(0) &= 1 - \mathcal{D}'(p_0) - \mathcal{D}'(p_2).\end{aligned}$$

  Then $\mathcal{D}'$ forms a feasible solution to the linear program (3.1), with support in $\{0\} \cup [p_0, 1]$, and with value $\texttt{LP}(\mathcal{D}', \mu) \geq \texttt{OPT}_{\texttt{LP}} - \Lambda p_0 n/p_2 \geq \texttt{OPT}_{\texttt{LP}} - \Lambda^2 p_0 n^2/B$.

□

Let ALG be a BwK algorithm which achieves regret bound (1.1). Suppose ALG is applied to a discretized instance of dynamic procurement, with m arms, n agents and budget B. For unit supply ($\Lambda = 1$), ALG has regret $R(m, n, B) = \tilde{O}(\sqrt{mn} + n\sqrt{m/B})$. For non-unit supply, in order to apply ALG we need to normalize the problem instance so that per-round rewards and consumptions are at most 1. That is, we need to divide all rewards, consumptions, and the budget, by the factor of Λ. Thus we obtain regret $R(m, n, B/\Lambda)$ on the normalized instance, and therefore regret $\Lambda R(m, n, B/\Lambda)$ on the original problem instance. Using Lemma 3.8.3 and Claim 3.8.4 (and normalizing / re-normalizing) we obtain the following regret bound:

Corollary 3.8.5. *Consider dynamic procurement with non-unit supply. Fix $\epsilon, p_0 \in (0, 1)$, and let S be the hyperbolic ϵ-mesh on $[p_0, 1]$. Then running* ALG *on arms $S \times \{1, \ldots, \Lambda\}$ achieves regret*

$$\Lambda R(\Lambda|S|, n, B/\Lambda) + \epsilon B + p_0 \Lambda^2 n^2/B.$$

Optimizing the parameters ϵ, p_0 in Corollary 3.8.5, we obtain the final regret bound.

Theorem 3.8.6. *Consider dynamic procurement with non-unit supply. Assume n agents, budget B and ceiling Λ. Let* ALG *be a* BwK *algorithm which achieves regret bound (1.1). Applying* ALG *with a suitably chosen discretization yields regret $\tilde{O}(\Lambda^{3/2}\, n/B^{1/4})$.*

3.9 Optimal dynamic policy beats the best fixed arm

Let us provide some examples of BwK problem instances in which the best dynamic policy (in fact, the best fixed distribution over arms) beats the best fixed arm. We start with a very simple (but perhaps contrived) example of this phenomenon. Then we provide more realistic examples that arise in the domains of dynamic pricing and dynamic procurement.

Simple example. There are d arms; pulling arm i deterministically produces a reward of 1, consumes one unit of resource i, and does not consume any other resources. We are given an initial endowment of B units of each resource. Any policy that plays a fixed arm i in each period is limited to a total reward of B before running out of its budget of resource i. A policy that always plays a uniformly random arm achieves an expected reward of nearly $d \cdot B$ before its resource budget runs out.

Dynamic pricing. Consider the basic setting of "dynamic pricing with limited supply": in each round a potential buyer arrives, and the seller offers him one item at a price; there are k items and $n > k$ potential buyers. One can easily con-

struct distributions for which offering a mixture of two prices is strictly superior to offering any fixed price. In fact this situation arises whenever the "revenue curve" (the mapping from prices to expected revenue) is non-concave and its value at the quantile k/n lies below its concave hull.

Let us provide a specific example. Fix any constant $\delta > 0$, and let $\epsilon = k^{\delta-1/2}$. Each buyer has the following two-point demand distribution: the buyer's value for item is $v = 1$ with probability $\frac{1}{n} k^{1/2+\delta}$, and $v = \epsilon$ with the remaining probability.

To analyze this example, let $\texttt{REW}(\mathcal{D})$ be the expected total reward (i.e., the expected total revenue) from using a fixed distribution $\mathcal{D}$ over prices in each round; let $\texttt{REW}(p)$ be the same quantity when $\mathcal{D}$ deterministically picks a given price p.

- Clearly, if one offers a fixed price in all rounds, it only makes sense to offer prices $p = \epsilon$ and $p = 1$. It is easy to see that $\texttt{REW}(\epsilon) = k \cdot \epsilon \leq k^{1/2+\delta}$. and $\texttt{REW}(1) = 1 \cdot \Pr[\text{sale at price 1 }] \leq k^{1/2+\delta}$.
- Now consider a distribution $\mathcal{D}$ which picks price ϵ with probability $\frac{k-k^{1/2+\delta}}{n}$, and picks price 1 with the remaining probability. It is easy to show that $\texttt{REW}(D) \geq (2 - o(1))\, k^{1/2+\delta}$.

So, $\texttt{REW}(\mathcal{D})$ is essentially twice as large compared to the total expected revenue of the best fixed arm.

Dynamic procurement. A similar example can be constructed in the domain of dynamic procurement. Consider the basic setting thereof: in each round a potential seller arrives, and the buyer offers to buy one item at a price; there are

T sellers and the buyer is constrained to spend at most budget B. The buyer has no value for left-over budget and each sellers value for the item is drawn i.i.d from an unknown distribution. Then a mixture of two prices is strictly superior to offering any fixed price whenever the "sales curve" (the mapping from prices to probability of selling) is non-concave and its value at the quantile B/T lies below its concave hull.

Let us provide a specific example. Fix any constant $\delta > 0$, and let $\epsilon = B^{1/2+\delta}$. Each seller has the following two-point demand distribution: the seller's value for item is $v = 1$ with probability $\frac{B}{T}$, and $v = 0$ with the remaining probability. We use the notation $\texttt{REW}(\mathcal{D})$ and $\texttt{REW}(p)$ as defined above.

- Clearly, if one offers a fixed price in all rounds, it only makes sense to offer prices $p = 0$ and $p = 1$. It is easy to see that $\texttt{REW}(0) <= T \cdot \mathbb{E}[\text{Probability of selling at price 0}] = B$. and $\texttt{REW}(1) = B$.
- Now consider a distribution $\mathcal{D}$ which picks price 0 with probability $1 - \frac{B-\epsilon}{T}$, and picks price 1 with the remaining probability. It is easy to show that $\texttt{REW}(D) \geq (2 - o(1))\, B$.

Again, $\texttt{REW}(\mathcal{D})$ is essentially twice as large compared to the total expected sales of the best fixed arm.

CHAPTER 4

RESOURCE CONSTRAINED CONTEXTUAL BANDITS

In this chapter we consider the generalization of the BwK problem which we call "Resourceful contextual bandits". In the previous chapter we introduced two algorithms for the BwK problem and we generalize one of them to get an algorithm for the contextual version in this chapter.

In section 4.1 we introduce the tools needed for our result. Then in section 4.2 we will give will generalize the algorithm Mixture_Elimintation for BwK to Contexutal_Mixture_Elimination for the contextual version and present its analysis in sections 4.3, 4.4 and 4.8. We show that our bounds are tight in some regimes with a lowerbound in section 4.6. We detail some of the discretization issue for certain applications in section 4.5 and discuss about the applications in detail in section 4.7.

4.1 Tools

4.1.1 Linear approximation and the benchmark

We set up a linear relaxation that will be crucial throughout the paper. As a by-product, we (effectively) reduce our benchmark to the best fixed distribution over policies.

A given distribution P over policies defines an algorithm $\texttt{ALG}_P$: in each round a policy π is sampled independently from P, and the action $a = \pi(x)$ is chosen. The *value* of P is the total reward of this algorithm, in expectation over the out-

come distribution.

As the value of P is difficult to characterize exactly, we approximate it (generalizing the approach from [7, 10] for the non-contextual version). We use a linear approximation where all rewards and consumptions are deterministic and the time is continuous. Let $r(P,\mu)$ and $c_i(P,\mu)$ be the expected per-round reward and the expected per-round consumption of resource i for policy $\pi \sim P$, given expected-outcomes tuple μ. Then the linear approximation corresponds to the solution of a simple linear program:

$$\begin{array}{lrcll} \text{Maximise} & t\, r(P,\mu) & & & \text{in } t \in \mathbb{R} \\ \text{subject to} & t\, c_i(P,\mu) & \leq & B & \text{for each } i \\ & t & \geq & 0. & \end{array} \tag{4.1}$$

The solution to this LP, which we call the *LP-value* of P, is

$$\texttt{LP}(P,\mu) = r(P,\mu)\ \min_i B/c_i(P,\mu). \tag{4.2}$$

Denote $\texttt{OPT}_{\texttt{LP}} = \sup_P \texttt{LP}(P,\mu)$, where the supremum is over all distributions P over Π.

Lemma 4.1.1. $\texttt{OPT}_{\texttt{LP}} \geq \texttt{OPT}(\Pi)$.

Therefore, it suffices to compete against the best fixed distribution over Π (as approximated by $\texttt{OPT}_{\texttt{LP}}$), even though our benchmark $\texttt{OPT}(\Pi)$ allows unrestricted changes over time.

A distribution P over Π that attains the supremum value $\texttt{OPT}_{\texttt{LP}}$ is called *LP-optimal*. Such P is called *LP-perfect* if furthermore $|\texttt{support}(P)| \leq d+1$ and $c_i(P,\mu) \leq B/T$ for each resource i. We find it useful to consider LP-perfect distributions throughout the paper.

Lemma 4.1.2. *An LP-perfect distribution exists for any instance of* RCB.

Lemma 4.1.1 and Lemma 4.1.2 hold for the non-contextual version of RCB [10]. The general case can be reduced to the non-contextual version via a standard reduction where actions in the new problem correspond to policies in Π in the original problem.

4.2 The algorithm: Contexutal_Mixture_Elimination

The algorithm's goal is to converge on a LP-perfect distribution over policies. The general design principle is to explore as much as possible while avoiding obviously suboptimal decisions.

Overview of the algorithm. In each round t, the following happens.

1. *Compute estimates.* We compute high-confidence estimates for the per-round reward $r(\pi)$ and per-round consumption $c_i(\pi)$, for each policy $\pi \in \Pi$ and each resource i. The collection $\mathcal{I}$ of all expected-outcomes tuple that are consistent with these high-confidence estimates is called the *confidence region*.

2. *Avoid obviously suboptimal decisions.* We prune away all distributions P over policies in Π that are not LP-perfect with high confidence. More precisely, we prune all P that are not LP-perfect for any expected-outcomes tuple in the confidence region $\mathcal{I}$; the remaining distributions are called *potentially LP-perfect*. Let $\mathcal{F}$ be the convex hull of the set of all potentially LP-perfect distributions.

3. *Explore as much as possible.* We choose a distribution $P \in \mathcal{F}$ which is *balanced*,

in the sense that no action is starved; see Equation (4.3) for the precise definition. Note that balanced distributions are typically *not* LP-perfect.

4. *Select an action.* We choose policy $\pi \in \Pi$ independently from P. Given context x, the action a is chosen as $a = \pi(x)$. The algorithm adds some random noise: with probability q_0, the action a is instead chosen uniformly at random, for some parameter q_0.

The algorithm halts as soon as the time horizon is met, or one of the resources is exhausted.

The pseudocode can be found in Algorithm 5.

Some details. After each round t, we estimate the per-round consumption $c_i(\pi)$ and the per-round reward $r(\pi)$, for each policy $\pi \in \Pi$ and each resource i, using the following unbiased estimators:

$$\widehat{c_i}(\pi) = \frac{c_i \, \mathbf{1}_{\{a=\pi(x)\}}}{P[a = \pi(x) \,|\, x]} \text{ and } \widehat{r}(\pi) = \frac{r \, \mathbf{1}_{\{a=\pi(x)\}}}{P[a = \pi(x) \,|\, x]}.$$

The corresponding time-averages up to round t are denoted

$$\hat{c}_{t,i}(\pi) = \tfrac{1}{t-1} \sum_{s=1}^{t-1} \widehat{c}_{s,i}(\pi) \text{ and } \hat{r}_t(\pi) = \tfrac{1}{t-1} \sum_{s=1}^{t-1} \widehat{r}_s(\pi).$$

We show that with high probability these time-averages are close to their respective expectations. To express the confidence term in a more lucid way, we use the following shorthand, called *confidence radius*: $\texttt{rad}_t(\nu) = \sqrt{C_{\texttt{rad}} \, \nu / t}$, where $C_{\texttt{rad}} = \Theta(\log(dK\,T\,|\Pi|))$ is a parameter which we will fix later. We show that w.h.p. the following holds:

$$|r(\pi) - \hat{r}_t(\pi)| \leq \texttt{rad}_t\,(K/\alpha_\pi)\,, \tag{4.4}$$

$$|c_i(\pi) - \hat{c}_{t,i}(\pi)| \leq \texttt{rad}_t\,(\,K/\alpha_\pi) \quad \text{for all } i. \tag{4.5}$$

Algorithm 5: `Contexutal_Mixture_Elimination`

Parameters: #actions K, time horizon T, budget B,
benchmark set Π, context distribution $\mathcal{D}_{\mathtt{X}}$.

Data structure: "confidence region"
$\mathcal{I} \leftarrow \{$ all feasible expected-outcomes tuples $\}$.

For each round $t = 1 \dots T$ **do**

  $\mathbf{\Delta} = \{$distributions P over Π: P is LP-perfect for some $\mu' \in \mathcal{I}\}$.

  Let $\mathcal{F}$ be the convex hull of $\mathbf{\Delta}$.

  Let $\alpha_\pi = \max_{P \in \mathcal{F}} P(\pi), \quad \forall \pi \in \Pi$.

  Choose a "balanced" distribution $P \in \mathcal{F}$: any P such that $\forall \pi \in \Pi$

$$\mathop{\mathbb{E}}_{x \sim \mathcal{D}_{\mathtt{X}}} \left[\frac{1}{(1-q_0)\, P(\pi(x)|x) + \frac{q_0}{K}} \right] \leq \frac{2K}{\alpha_\pi}, \text{ where } q_0 = \min\left(\tfrac{1}{2}, \sqrt{\tfrac{K}{T} \log(K\,T|\Pi|)}\right). \tag{4.3}$$

  Observe context x; **choose** action a to "play":

    with probability q_0, draw a u.a.r. in A; else, draw $\pi \sim P$ and let $a = \pi(x)$.

  Observe outcome vector $(r, c_1, \dots, c_d)$.

  Halt if one of the resources is exhausted.

  Eliminate expected-outcomes tuples from $\mathcal{I}$ according to
  equations (4.4-4.5)

(Here $\alpha_\pi = \max_{P \in \mathcal{F}} P(\pi)$, as in Algorithm 5.)

4.3 Correctness of the algorithm

We need to prove that in each round t, some $P \in \mathcal{F}$ satisfies (4.3), and Equations (4.4-4.5) hold for all policies $\pi \in \Pi$ with high probability.

Notation. Let $\alpha_{\pi,t}$ denote the α_π, $\pi \in \Pi$ computed in round t, and let P_t be the distribution over Π chosen in this round. Let q_0 be the noise probability. The "noisy version" of P_t is defined as

$$P'_t(a|x) = (1 - q_0)\, P_t(a|x) + q_0/K \qquad (\forall x \in X, a \in A).$$

Then action a in round t is drawn from distribution $P'_t(\cdot|x_t)$.

Lemma 4.3.1. *In each round t, some $P \in \mathcal{F}$ satisfies (4.3).*

Proof. We extend the minimax argument from [23]. Let $\mathcal{F}_\Pi$ be the set of all distributions over Π. Our proof works for any $q_0 \in [0, \frac{1}{2}]$ and any compact and convex set $\mathcal{F} \subset \mathcal{F}_\Pi$.

Equation (4.3) holds for a given $P \in \mathcal{F}$ if and only if for every distribution $Z \in \mathcal{F}_\Pi$ we have that

$$f(P, Z) \triangleq \mathop{\mathbb{E}}_{x\sim\mathcal{D}_X} \mathop{\mathbb{E}}_{\pi\sim Z} \left[\frac{\alpha_\pi}{P'(\pi(x)|x)} \right] \leq 2K,$$

where P' is the noisy version of P. It suffices to show that

$$\min_{P\in\mathcal{F}} \max_{Z\in\mathcal{F}_\Pi} f(P, Z) \leq 2K. \tag{4.6}$$

We use a min-max argument: noting that f is a convex function of P and a concave function of Z, by the Sion's min-max theorem we have that

$$\min_{P\in\mathcal{F}} \max_{Z\in\mathcal{F}_\Pi} f(P, Z) = \max_{Z\in\mathcal{F}_\Pi} \min_{P\in\mathcal{F}} f(P, Z). \tag{4.7}$$

For each policy $\pi \in \Pi$, let $\beta_\pi \in \operatorname{argmax}_{\beta \in \mathcal{F}} \beta(\pi)$ be a distribution which maximizes the probability of selecting π. Such distribution exists because $\beta \mapsto \beta(\pi)$ is a continuous function on a convex set $\mathcal{F}$. Recall that $\alpha_\pi = \beta_\pi(\pi)$.

Given any $Z \in \mathcal{F}_\Pi$, define distribution $P_Z \in \mathcal{F}_\Pi$ by $P_Z(\pi) = \sum_{\phi \in \Pi} Z(\phi)\beta_\phi(\pi)$. Note that P_Z is a convex combination of distributions in $\mathcal{F}$. Since $\mathcal{F}$ is convex, it follows that $P_Z \in \mathcal{F}$. Also, note that $P_Z(a|x) \geq \sum_{\pi \in \Pi:\ \pi(x)=a} Z(\pi)\,\alpha_\pi$. Letting P'_Z be the noisy version of P_Z, we have:

$$\begin{aligned}
\min_{P \in \mathcal{F}} f(P, Z) \leq f(P_Z, Z) &= \mathop{\mathbb{E}}_{x \sim \mathcal{D}_X} \left[\sum_\pi \frac{Z(\pi)\,\alpha_\pi}{P'_Z(\pi(x)|x)} \right] \\
&= \mathop{\mathbb{E}}_{x \sim \mathcal{D}_X} \left[\sum_{a \in A} \sum_{\pi \in \Pi:\ \pi(x)=a} \frac{Z(\pi)\,\alpha_\pi}{P'_Z(a|x)} \right] = \mathop{\mathbb{E}}_{x \sim \mathcal{D}_X} \left[\sum_{a \in X} \frac{\sum_{\pi \in \Pi:\ \pi(x)=a} Z(\pi)\,\alpha_\pi}{(1-q_0)P'_Z(a|x) + q_0/K} \right] \\
&\leq \mathop{\mathbb{E}}_{x \sim \mathcal{D}_X} \left[\sum_{a \in X} \frac{1}{1-q_0} \right] = \frac{K}{1-q_0} \leq 2K.
\end{aligned}$$

Thus, by Equation (4.7) we obtain Equation (4.6). □

To analyze Equations (4.4-4.5), we will use concentration bound stated in lemma 2.2.4.

Lemma 4.3.2. *With probability at least* $1 - \frac{1}{T}$*, Equations (4.4-4.5) hold for all rounds* t *and policies* $\pi \in \Pi$.

Proof. Let us prove Equation (4.4). (The proof of (4.5) is similar.) Fix round t and policy $\pi \in \Pi$. We bound the conditional variance of the estimators $\widehat{r}_t(\pi)$. Specifically, let $\mathcal{G}_t$ be the σ-algebra induced by all events up to (but not including) round t. Then

$$\mathbb{E}\left[\widehat{r}_t(\pi)^2 \,|\, \mathcal{G}_t\right] = \mathop{\mathbb{E}}_{x \sim \mathcal{D}_X,\ a \sim P'_t} \left[\frac{r_t^2\ \mathbf{1}_{\{\pi(x)=a\}}}{P'_t(a|x)^2} \right] \leq \mathop{\mathbb{E}}_{x \sim \mathcal{D}_X} \left[\frac{1}{P'_t(\pi(x)|x)} \right] \leq \frac{2K}{\alpha_{\pi,t}}.$$

The last inequality holds by the algorithm's choice of distribution P_t. Since the confidence region $\mathcal{I}$ in our algorithm is non-increasing over time, it follows that

$\alpha_{\pi,t}$ is non-increasing in t, too. We conclude that $\texttt{Var}\left[\widehat{r}_s(\pi) \,|\, \mathcal{G}_t\right] \leq 2K/\alpha_{\pi,t}$ for each round $s \leq t$. Therefore, noting that $\widehat{r}_t(\pi) \leq 1/P'(\pi(x_t)|x_t) \leq K/q_0$, we obtain Equation (4.4) by applying Lemma 2.2.4 with $X_t = \widehat{r}_t(\pi) - r(\pi)$. □

4.4 Regret analysis: proof of Theorem 1.5.4

We provide the key steps of the proof; the remaining pieces can be found in Section 4.8.

Let $\mathcal{I}_t$ and $\mathbf{\Delta}_t$ be respectively, the confidence region $\mathcal{I}$ and the set $\mathbf{\Delta}$ of potentially LP-perfect distributions computed in round t. Let $\texttt{Conv}(\mathbf{\Delta}_t)$ be the convex hull of $\mathbf{\Delta}_t$.

First we bound the deviations within the confidence region.

Lemma 4.4.1. *For any two expected-outcomes tuples $\mu', \mu'' \in \mathcal{I}_t$ and a distribution $P \in \texttt{Conv}(\mathbf{\Delta}_t)$:*

$$|c_i(P,\mu') - c_i(P,\mu'')| \leq \texttt{rad}_t(dK) \qquad \text{for each resource } i \tag{4.8}$$

$$|r(P,\mu') - r(P,\mu'')| \leq \texttt{rad}_t(dK) \tag{4.9}$$

Proof. Let us prove Equation (4.9). (Equation (4.8) is proved similarly.) By definition of $\mathcal{I}_t$:

$$\begin{aligned} |r(P,\mu') - r(P,\mu'')| &\leq \textstyle\sum_{\pi\in\Pi} P(\pi)\, |r(\pi,\mu') - r(\pi,\mu'')| \\ &\leq \textstyle\sum_{\pi\in\Pi} P(\pi)\, \texttt{rad}_t\left(K/\alpha_{\pi,t}\right). \end{aligned}$$

It remains to prove that the right-hand side is at most $\texttt{rad}_t(dK)$. By linearity, it suffices to prove this for $P \in \mathbf{\Delta}_t$. So let us assume $P \in \mathbf{\Delta}_t$ from here on. Recall

that $|\texttt{support}(P)| \leq d$ since P is LP-perfect, and $P(\pi) \leq \alpha_{\pi,t}$ for any policy $\pi \in \Pi$. Therefore:

$$\begin{aligned}\sum_{\pi\in\Pi} P(\pi)\ \texttt{rad}_t\left(K/\alpha_{\pi,t}\right) &\leq \sum_{\pi\in\Pi}\ \texttt{rad}_t\left(KP(\pi)\right)\\ &\leq \texttt{rad}_t\left(dK\sum_{\pi\in\Pi} P(\pi)\right) = \texttt{rad}_t\left(dK\right).\end{aligned}$$ □

Using Lemma 4.4.1 and a long computation (fleshed out in Section 4.8), we prove the following two lemmas.

Lemma 4.4.2. *For any two expected-outcomes tuples $\mu', \mu'' \in \mathcal{I}_t$ and a distribution $P \in \texttt{Conv}(\mathbf{\Delta}_t)$:*

$$\texttt{LP}(P,\mu') - \texttt{LP}(P,\mu'') \leq (\tfrac{1}{B}\,\texttt{LP}(P,\mu') + 2)\cdot T\cdot \texttt{rad}_t(dK).$$

Lemma 4.4.3. *For any distribution $P' \in \texttt{Conv}(\mathbf{\Delta}_t)$ and any expected-outcomes tuple $\mu \in \mathcal{I}_t$,*

$$\min_{P\in\mathbf{\Delta}_t} \texttt{LP}(P,\mu) \leq \texttt{LP}(P',\mu) \leq \max_{P\in\mathbf{\Delta}_t} \texttt{LP}(P,\mu). \tag{4.10}$$

Proof. The proof consists of two parts. The second inequality in Equation (4.10) follows easily because the distribution which maximizes $\texttt{LP}(P,\mu)$ by definition belongs to $\mathbf{\Delta}_t$, and so

$$\texttt{LP}(P',\mu) \leq \max_{P\in\texttt{Conv}(\mathbf{\Delta}_t)} \texttt{LP}(P,\mu) = \max_{P\in\mathbf{\Delta}_t} \texttt{LP}(P,\mu).$$

To prove the first inequality in Equation (4.10), we first argue that $\texttt{LP}(P,\mu)$ is a quasi-concave function of P. Denote $\eta_i(P,\mu) = B\cdot r(P,\mu)/c_i(P,\mu)$ for each resource i. Then η_i is a quasi-concave function of P since each *level set* (the set of distributions P that satisfy $\eta_i(P,\mu) \geq \alpha$ for some $\alpha \in \mathbb{R}$) is a convex set. Therefore $\texttt{LP}(P,\mu) = \min_i \eta_i(P,\mu)$ is a quasi-concave function of P as a minimum of quasi-concave functions.

Since $P' \in \texttt{Conv}(\mathbf{\Delta}_t)$, it is a convex combination $P' = \sum_{Q\in\mathbf{\Delta}_t} \alpha_Q\ Q$ with $\sum_{Q\in\mathbf{\Delta}_t} \alpha_Q = 1$. Therefore:

$$\begin{aligned}
\texttt{LP}(P',\mu) &= \texttt{LP}\left(\sum_{Q\in\mathbf{\Delta}_t} \alpha_Q\ Q,\ \mu\right) \\
&\geq \min_{Q\in\mathbf{\Delta}_t,\alpha_Q>0} \texttt{LP}(Q,\mu) \qquad \text{By definition of quasi-concave functions} \\
&\geq \min_{Q\in\mathbf{\Delta}_t} \texttt{LP}(Q,\mu). \qquad \square
\end{aligned}$$

Let $\texttt{REW}_t$ and $C_{t,i}$ be, respectively, the (realized) total reward and average consumption of resource i up to and including round t. Recall that P'_t is the noisy version of distribution P_t chosen by the algorithm in round t. Given P_t, the expected revenue and resource-i consumption in round t is, respectively, $r(P'_t,\mu)$ and $c_i(P'_t,\mu)$. Denote $\overline{r}_t = \frac{1}{t}\sum_{i=1}^{t} r(P'_t,\mu)$ and $\overline{c}_{i,t} = \frac{1}{t}\sum_{i=1}^{t} c_i(P'_t,\mu)$.

Henceforth we assume a ***clean execution*** where several high-probability conditions are satisfied. Formally, the algorithm's execution is *clean* if in each round t Equations (4.4-4.5) are satisfied, and moreover $\min\left(|\frac{1}{t}\,\texttt{REW}_t - \overline{r}_t|,\ |C_{t,i} - \overline{c}_{t,i}|\right) \leq \texttt{rad}_t(1)$.

In particular, the set $\mathbf{\Delta}_t$ of potentially LP-perfect distributions indeed contains a LP-perfect distribution. By Lemma 4.3.2 and Azuma-Hoeffding Inequality, clean execution happens with probability at least $1 - \frac{1}{T}$. Thus, it suffices to lower-bound the total reward $\texttt{REW}_T$ for a clean execution.

The following lemma captures a crucial argument. Denote

$$\begin{aligned}
\Phi_t &= \left(2 + \tfrac{1}{B}\left[\max_{P\in\mathcal{F}_\Pi,\ \mu\in\mathcal{I}_t} \texttt{LP}(P,\mu)\right]\right)\cdot T\cdot \texttt{rad}_t(dK) \\
\Psi_t &= (2 + \tfrac{1}{B}\,\texttt{OPT}_{\texttt{LP}})\cdot T\cdot \texttt{rad}_t(dK).
\end{aligned}$$

Lemma 4.4.4. *For any expected-outcomes tuple $\mu^*, \mu^{**} \in \mathcal{I}_t$ and distributions $P', P'' \in$*

$\texttt{Conv}(\mathbf{\Delta}_t)$:

$$|\text{LP}(P', \mu^*) - \text{LP}(P'', \mu^{**})| \leq 3\Phi_t. \tag{4.11}$$

Proof. Assume $P', P'' \in \mathbf{\Delta}_t$. In particular, P', P'' are LP-perfect for some expected-outcomes tuples $\mu', \mu'' \in \mathcal{I}_t$, respectively. Also, some distribution $P^* \in \mathbf{\Delta}_t$ is LP-perfect for μ^*. Therefore:

$$\begin{aligned}
\text{LP}(P', \mu^*) &\geq \text{LP}(P', \mu') - \Phi_t && \text{(by Lemma 4.4.2: } P = P') \\
&\geq \text{LP}(P^*, \mu') - \Phi_t \\
&\geq \text{LP}(P^*, \mu^*) - 2\Phi_t && \text{(by Lemma 4.4.2: } P = P^*) \\
&\geq \text{LP}(P'', \mu^*) - 2\Phi_t.
\end{aligned}$$

We proved Equation (4.11) for $P', P'' \in \mathbf{\Delta}_t$. Thus:

$$\min_{P \in \mathbf{\Delta}_t} \text{LP}(P, \mu^*) - \max_{P \in \mathbf{\Delta}_t} \text{LP}(P, \mu^*) \leq 2\Phi_t. \tag{4.12}$$

Next we generalize to $P', P'' \in \texttt{Conv}(\mathbf{\Delta}_t)$.

$$\begin{aligned}
\text{LP}(P', \mu^*) &\geq \min_{P \in \mathbf{\Delta}_t} \text{LP}(P, \mu^*) && \text{(by Lemma 4.4.3)} \\
&\geq \max_{P \in \mathbf{\Delta}_t} \text{LP}(P, \mu^*) - 2\Phi_t && \text{(by Equation (4.12))} \\
&\geq \text{LP}(P'', \mu^*) - 2\Phi_t && \text{(by Lemma 4.4.3).}
\end{aligned}$$

We proved Equation (4.11) for $\mu^* = \mu^{**}$. We obtain the general case by plugging in Lemma 4.4.2. □

Corollary 4.4.5. $\Phi_t \leq 2\Psi_t$, *assuming that* $B \geq 6 \cdot T \cdot \texttt{rad}_t(dK)$.

Proof. Follows from Lemma 4.4.4 via a simple computation. □

Corollary 4.4.6. $\texttt{LP}(P_t, \mu) \geq \texttt{OPT}_{\texttt{LP}} - 12\,\Psi_t$, *where* μ *is the actual expected-outcomes tuple.*

Proof. Follows from Lemma 4.4.4 and Corollary 4.4.5, observing that $P_t \in \texttt{Conv}(\mathbf{\Delta}_t)$ and $\texttt{OPT}_{\texttt{LP}} = \texttt{LP}(P^*, \mu)$ for some $P^* \in \mathbf{\Delta}_t$. □

In the remainder of the proof (which is fleshed out in Section 4.8) we build on the above lemmas and corollaries to prove the following sequence of claims:

$$\begin{aligned} \texttt{REW}_t &\geq \frac{t}{T}\,(\texttt{OPT}_{\texttt{LP}} - O(\Psi_t)) \\ C_{t,i} &\leq B/T + O(\texttt{rad}_t(dK)) \\ \texttt{REW}_T &\geq \texttt{OPT}_{\texttt{LP}} - O(\Psi_T). \end{aligned} \tag{4.13}$$

To complete the proof of Theorem 1.5.4, we re-write the last equation as $\texttt{REW}_T \geq f(\texttt{OPT}_{\texttt{LP}})$ for an appropriate function $f()$, and observe that $f(\texttt{OPT}_{\texttt{LP}}) \geq f(\texttt{OPT})$ because function $f()$ is increasing.

4.5 Discretization issues

In order to immediately apply Theorem 1.5.4, the set of all actions used by policies in Π, denote it $A(\Pi)$, should be finite and small compared to T. However, in some applications, such as dynamic pricing and dynamic procurement, the set of all possible actions is infinite, so $A(\Pi)$ may be prohibitively large or infinite, too. One way to handle such problem instances is *discretization*. Below,

we outline the general approach, and work out the paradigmatic special case of contextual dynamic pricing with a single product.

In particular, we obtain the following corollary:

Corollary 4.5.1. *Consider contextual dynamic pricing with a limited supply B of a single product. Let Π be the policy set and T be the time horizon. Then there exists a policy set Π' such that algorithm* `Contexutal_Mixture_Elimination` *satisfies*

$$\texttt{REW}(\Pi') \geq \texttt{OPT}(\Pi) - O\left((TB)^{1/3} \cdot \log\left(TB|\Pi|\right)\right). \tag{4.14}$$

Here $\texttt{REW}(\Pi')$ *is the algorithm's total expected reward if it is given policy set Π'.*

Note that if B is at least a constant fraction of T, we obtain regret $\tilde{O}(T^{2/3})$, which matches the lower bound (even) for the non-contextual version [7]. However, the optimal regret for the non-contextual version is $\tilde{O}(B^{2/3})$ regret, and we do not match that. With an upper bound of $\tilde{O}(BT^{1/3})$ and a lower bound of $\Omega(B^{2/3})$, it is not clear what is the optimal regret here.

The general approach is as follows. For each $\epsilon > 0$ we define a ϵ-discretized policy set Π_ϵ that approximates Π so that $|A(\Pi_\epsilon)|$ decreases with ϵ. The ϵ controls the tradeoff between the learning rate and the discretization error. We use Theorem 1.5.4 to obtain a performance guarantee that depends on ϵ. Further, we need a separate result that upper-bounds the discretization error of Π_ϵ compared to Π. Then we can optimize the choice of ϵ to provide an algorithm with a specific regret bound.

In what follows we apply this approach to dynamic pricing with B copies of a single product. Recall that in this setting actions are prices, and the action set is $[0, 1] \cup \{\infty\}$, the set of all possible prices. Here $p = \infty$ is the special price that corresponds to skipping a round.

First, let us introduce some notation. Let Π be an arbitrary (finite) policy set. Let $S(p|x)$ be the probability of a sale for price p and context x. For a randomized policy π, define $S(\pi|x) = \mathbb{E}_{p\sim\pi(x)}[\, S(p|x)\,]$. Let $f_\epsilon(p)$, $p \geq 0$ be the largest price $p' \leq p$ such that $p' \in \epsilon\mathbb{N}$.

Second, for each policy $\pi \in \Pi$ we define a randomized policy π_ϵ: for each context x,

$$\pi_\epsilon(x) = \begin{cases} f_\epsilon(\pi(x)) & \text{with probability } \frac{S(\pi|x)}{S(f_\epsilon(\pi(x))|x)} \\ \infty & \text{with the remaining probability.} \end{cases} \tag{4.15}$$

The ϵ-discretized policy set is defined as $\Pi_\epsilon = \{\pi_\epsilon : \pi \in \Pi\}$.

Observe that the π_ϵ satisfies the following useful properties:

(i) $A(\pi_\epsilon) \subset \epsilon\mathbb{N} \cap [0,1]$, so that $|A(\Pi_\epsilon)| \leq \frac{1}{\epsilon}$.

(ii) $\pi(x) \geq \pi_\epsilon(x) \geq \pi(x) - \epsilon$ for all contexts x,

(iii) $S(\pi_\epsilon|x) = S(\pi|x)$ for all contexts x.

Third, let us bound the discretization error.

Lemma 4.5.2. $\texttt{OPT}_{\texttt{LP}}(\Pi) - \texttt{OPT}_{\texttt{LP}}(\Pi_\epsilon) \leq \epsilon B$, *for each* $\epsilon > 0$.

Proof. Fix $\epsilon > 0$ and a distribution P over policies in Π. Define the ϵ-discretized distribution P_ϵ over Π_ϵ in a natural way: for each policy $\pi \in \Pi$, select π_ϵ with probability $P(\pi)$. It suffices to prove that $\texttt{LP}(P_\epsilon) \geq \texttt{LP}(P) - \epsilon B$.

Let $r(P)$ and $c(P)$ denote the expected per-round reward and the expected

per-round consumption for distribution P. Then

$$\begin{aligned}
c(P_\epsilon) &= \mathop{\mathbb{E}}_{x,\pi}\left[\, S(\pi_\epsilon|x)\,\right] = \mathop{\mathbb{E}}_{x,\pi}\left[\, S(\pi_\epsilon|x)\,\right] = c(P)\\
r(P_\epsilon) &= \mathop{\mathbb{E}}_{x,\pi}\left[\, f_\epsilon(\pi(x))\cdot S(\pi_\epsilon|x)\,\right]\\
&\geq \mathop{\mathbb{E}}_{x,\pi}\left[\, (\pi(x)-\epsilon)\cdot S(\pi_\epsilon|x)\,\right]\\
&\geq \mathop{\mathbb{E}}_{x,\pi}\left[\, \pi(x)\cdot S(\pi|x)\,\right] - \epsilon \mathop{\mathbb{E}}_{x,\pi}\left[\, S(\pi_\epsilon|x)\,\right]\\
&= r(P) - \epsilon\, c(P_\epsilon).\\
\texttt{LP}(P_\epsilon) &= r(P_\epsilon)\,\frac{B}{c(P_\epsilon)} \geq r(P)\,\frac{B}{c(P)} - \epsilon B.\\
&\geq \texttt{LP}(P) - \epsilon B. \qquad \square
\end{aligned}$$

Fourth, let us formulate an appropriate version Theorem 1.5.4. Recall that we actually prove a somewhat stronger statement that is with respect to $\texttt{OPT}_{\texttt{LP}}(\Pi)$ rather than $\texttt{OPT}(\Pi)$. Namely, the algorithm's expected total reward $\texttt{REW}(\Pi)$ satisfies

$$\texttt{REW}(\Pi) \geq \texttt{OPT}_{\texttt{LP}}(\Pi) - O\left(1 + \tfrac{1}{B}\,\texttt{OPT}_{\texttt{LP}}(\Pi)\right)\sqrt{dKT\,\log\left(dKT\,|\Pi|\right)}.$$

For this particular setting we have $d = 1$ (single constraint) and $\texttt{OPT}_{\texttt{LP}}(\Pi) \leq B$, which implies

$$\texttt{REW}(\Pi) \geq \texttt{OPT}_{\texttt{LP}}(\Pi) - O\left(\sqrt{KT\,\log\left(KT\,|\Pi|\right)}\right).$$

Finally, plugging in Lemma 4.5.2, we obtain regret

$$\texttt{REW}(\Pi_\epsilon) \geq \texttt{OPT}_{\texttt{LP}}(\Pi) - \epsilon B - O\left(\sqrt{\tfrac{T}{\epsilon}\,\log\left(\tfrac{T}{\epsilon}\,|\Pi_\epsilon|\right)}\right), \qquad \text{for each } \epsilon > 0. \tag{4.16}$$

We choose the ϵ so as to optimize Equation (4.16).

In particular, we obtain Corollary 4.5.1 if we plug in $|\Pi_\epsilon| \leq |\Pi|$ (rather than consider how $|\Pi_\epsilon|$ decreases with ϵ) and pick $\epsilon = (\frac{dT}{B^2})^{1/3}$.

4.6 Lower bound: proof of Theorem 1.5.5

We will use the following lemma (which follows from simple probability arguments).

Lemma 4.6.1. *Consider two collections of n balls $\mathcal{I}_1$ and $\mathcal{I}_2$, each numbered from 1 to n. Let $\mathcal{I}_1$ consists of all red balls, while $\mathcal{I}_2$ consist of $n-1$ red balls and 1 green ball (with labels chosen uniformly at random). In this setting, let an algorithm is given access to random samples from one of $\mathcal{I}_i$ with replacement. The algorithm is allowed to first look at the ball's number and then decide whether to inspect it's color. Then any algorithm $\mathcal{A}$ which with probability at least $1/2$ can distinguish between $\mathcal{I}_1$ and $\mathcal{I}_2$ must inspect color of at least $n/2$ balls in expectation.*

Let us define a family of problem instances as follows. Let the set of arms be $\{a_1, a_2, \ldots, a_k\}$. There are T/B different contexts labelled $\{x_1, ..., x_{T/B}\}$ and there is a uniform distribution over contexts. There are T $(K-1)/B$ policies. Let them be indexed as $\pi_{i,j}$ for $2 \leq i \leq K$ and $1 \leq j \leq T/B$. Define them as follows: $\pi_{i,j}(x_l) = a_i$ for $l = j$, and $\pi_{i,j}(x_l) = a_1$ for $l \neq j$.

There is just one resource constraint. Pulling arm a_1 always costs 0 and arm $a_i, i \neq 1$ always costs 1. Now consider the following problem instances:

- Let $\mathcal{F}_0$ be the instance in which every arm always gives a reward 0. Note that $\texttt{OPT}(\mathcal{F}_0) = 0$.

- Let $\mathcal{F}_{i,j}$ be the instance in which arm a_j on context x_j gives reward 1, otherwise every arm on every context gives reward 0. Note that in this case the optimal distribution over policies is just to follow $\pi_{i,j}$ and gets reward $\approx B$.

Now consider any algorithm $\mathcal{A}$ and let the expected number of times it pulls arm a_i be p_i on input $\mathcal{F}_0$. Let $i', i' \neq 1$ be the arm for which this is minimum. Then by simple linearity of expectation we get that $B \geq (K-1)p_{i'}$. It is also simple to see that for the algorithm to get a regret better than $\Omega(\texttt{OPT})$ it should be able to distinguish between $\mathcal{F}_0$ and $\mathcal{F}_{i',.}$ at least with probability $1/2$. From lemma 4.6.1 this can be done iff $p_{i'} \geq T/(2B)$. Combining the two equations we get $B \geq (K-1)T/(2B)$. Solving for B we get $B \geq \sqrt{KT}/2$.

4.7 RCB: applications and special cases

In this section we discuss the application domains of resource-constrained (contextual) bandits in more detail. We focus on the three main application domains: dynamic pricing, dynamic procurement, and dynamic ad allocation. A more extensive discussion of these and other application domains (in the non-contextual version) can be found in [10, 11].

Dynamic pricing with limited supply. The algorithm is a monopolistic seller with a limited inventory. In the basic version, there is a limited supply of identical items. In each round, a new customer arrives, the algorithm picks a price, and offers one item for sale at this price. The customer then either buys the item at this price, or rejects the offer and leaves. The "context" represents the available information about the current customer, such as demographics, location, etc. The probability of selling at a given price for a given context (a.k.a. the *demand distribution*) is fixed over time, but not known to the algorithm. The algorithm optimizes the revenue; it does not derive any utility from the left-over items.

We represent this problem as an instance of RCB as follows. "Actions" are the possible prices, and the "resource constraint" is the number of items. In each round, the outcome vector is a pair (reward, items sold); if the offered price is p, the outcome vector is $(p, 1)$ if there is a sale, and $(0, 0)$ otherwise.

Many generalizations of dynamic pricing have been studied in the literature. In particular, RCB subsumes a number of extensions. First, an algorithm can sell multiple items to the same customer, possibly with volume discounts or surcharges. Second, an algorithm can have multiple products for sale, with limited inventory of each. Third, it may be advantageous to offer bundles consisting of different products, possibly with non-additive pricing (mirroring the non-additive valuations of the customers).

Dynamic procurement on a budget. The algorithm is a monopolistic buyer with a limited budget. The basic version is as follows. In each round, a new customer arrives, the algorithm picks a price, and offers to buy one item at this price. Then the customer either accepts the offer and sells the item at this price, or rejects the offer and leaves. The "context" is the available information on the current customer. The probability of buying at a given price for a given context (a.k.a. the "supply distribution") is fixed over time, but not known to the algorithm. The algorithm maximizes the number of items bought; it has no utility for the left-over money.

An alternative interpretation is that the algorithm is a contractor which hires workers to perform tasks, e.g. in a crowdsourcing market. In each round, a new worker arrives, the algorithm picks a price, and offers the worker to perform one task for this price; the worker then either accepts and performs the task at this price, or rejects and leaves. The relevant "context" for a worker in a

crowdsourcing market may include, for example, age, location, language, and task preferences.

Here, "actions" correspond to the possible prices, and the "resource constraint" is the buyer's budget. In each round, the outcome vector is a pair (items bought, money spent); if the offered price is p, then the outcome vector is $(1, p)$ if the offer is accepted, and $(0, 0)$ otherwise.

Dynamic procurement is a rich problem space, both for buying items and for hiring workers (see [48] for a discussion of the application to crowdsourcing markets). In particular, RCB subsumes a number of extensions of this basic setting. First, the algorithm may offer several tasks to the same worker, possibly at a discount. Second, there may be multiple types of tasks, each having a different value for the contractor; moreover, there may be additional budget constraints on each task type, or on various *subsets* of task types. Third, a given worker can be offered a bundle of tasks, consisting of tasks of multiple types, possibly with non-additive pricing. Fourth, there is a way to model the presence of competition (other contractors).

Dynamic ad allocation with budgets. The algorithm is an advertising platform. In the basic version, there is a fixed collection of ads to choose from. In each round, a user arrives, and the algorithm chooses one ad to display to this user. The user either clicks on this ad, or leaves without clicking. The algorithm receives a payment if and only if the ad is clicked; the payment for a given ad is fixed over time and known to the algorithm. The "context" is the available information about the user and the page on which the ad is displayed. The click probability for a given ad and a given context is constant over time, but not known to the algorithm.

Each ad belongs to some advertiser (who is the one paying the algorithm when this ad is clicked). Each advertiser may own multiple ads, and has a budget constraint: a maximal amount of money that can be spent on all his ads. Moreover, an advertiser may specify additional budget constraints on various subsets of the ads. The algorithm maximizes its revenue; it derives no utility from the left-over budgets.

Here, "actions" correspond to ads, and each budget corresponds to a separate resource. In a round when the chosen ad a is clicked, the reward is the corresponding payment v, and the resource consumption is v for each budget that involves a, and 0 for all other budgets. If the ad is not clicked, the reward and the consumption of each resource is 0.

RCB also subsumes more advanced versions in which multiple non-zero outcomes are possible in each round. For example, the ad platform may record what happens *after* the click, e.g. the time spent on the page linked from the ad and whether this interaction has resulted in a sale.

4.8 Regret analysis: missing proofs

Here we provide the remaining pieces for the regret analysis in Section 4.4.

4.8.1 Proof of Lemma 4.4.2

We restate the lemma for convenience.

Lemma For any two expected-outcomes tuples $\mu', \mu'' \in \mathcal{I}_t$ and a distribution $P \in \texttt{Conv}(\mathbf{\Delta}_t)$:

$$\texttt{LP}(P, \mu') - \texttt{LP}(P, \mu'') \leq (\tfrac{1}{B}\,\texttt{LP}(P, \mu') + 2) \cdot T \cdot \texttt{rad}_t(dK).$$

Proof. For brevity, we will denote:

$$\begin{aligned} \texttt{LP}' = \texttt{LP}(P, \mu') \quad &\text{and} \quad \texttt{LP}'' = \texttt{LP}(P, \mu'') \\ r' = r(P, \mu') \quad &\text{and} \quad r'' = r(P, \mu'') \\ c_i' = c_i(P, \mu') \quad &\text{and} \quad c_i'' = c_i(P, \mu''). \end{aligned}$$

By symmetry, it suffices to prove the upper bound for $\texttt{LP}' - \texttt{LP}''$. Henceforth, assume $\texttt{LP}' > \texttt{LP}''$.

We consider two cases, depending on whether

$$T \leq B/c_i'' \quad \text{for all resources } i. \tag{4.17}$$

Case 1. Assume Equation (4.17) holds. Then $\texttt{LP}'' = T\ r''$. Therefore by Lemma 4.4.1

$$\texttt{LP}' - \texttt{LP}'' \leq T\ r' - T\ r'' \leq T\ \texttt{rad}_t(dK).$$

Case 2. Assume Equation (4.17) fails. Then $\texttt{LP}'' = B\ r''/c_i''$ for some resource i. We consider two subcases, depending on whether

$$T \leq B/c_j' \quad \text{for all resources } j. \tag{4.18}$$

Subcase 1. Assume Equation (4.18) holds. Then:

$$\texttt{LP}' = T\ r' \tag{4.19}$$

$$\texttt{LP}'' \leq T \cdot \min(r',\ r'') \leq \texttt{LP}' \tag{4.20}$$

Equation (4.20) follows from (4.19) and $\mathtt{LP}' > \mathtt{LP}''$.

For $\delta \in [0, c_i'')$, define

$$r(\delta) = r'' + \delta$$

$$c_i(\delta) = c_i'' - \delta$$

$$f(\delta) = B\, r(\delta)/c_i(\delta).$$

Then $f()$ is monotonically and continuously increasing function, with $f(\delta) \to \infty$ as $\delta \to c_i''$. For convenience, define $f(c_i'') = \infty$.

Let $\delta_0 = \min(c_i'', \mathtt{rad}_t(dK))$. By Lemma 4.4.1, we have $f(\delta_0) \geq Br'/c_i'$. Therefore:

$$f(0) = \mathtt{LP}'' < \mathtt{LP}' \leq Br'/c_i' \leq f(\delta_0).$$

Thus, by Equation (4.20), we can fix $\delta \in [0, \delta_0)$ such that $f(\delta) = T \cdot \min(r',\ r'')$.

$$
\begin{aligned}
\mathtt{LP}'' = B\,\frac{r''}{c_i''} &= B\,\frac{r(\delta)-\delta}{c_i(\delta)+\delta} \\
&\geq B\,\frac{r(\delta)-\delta}{c_i(\delta)}\left(1-\frac{\delta}{c_i(\delta)}\right). \\
f(\delta)-\mathtt{LP}'' &\leq \frac{B}{c_i(\delta)}\delta + B\frac{r(\delta)}{c_i(\delta)^2}\delta \\
&= \left(1+\frac{r(\delta)}{c_i(\delta)}\right)\frac{B\,\delta}{c_i(\delta)} \\
&= \left(1+\frac{f(\delta)}{B}\right)\frac{f(\delta)\,\delta}{r(\delta)} \\
&\leq \left(1+\frac{T\ r'}{B}\right)\frac{T\ r''\ \delta}{r(\delta)} \\
&\leq \left(1+\frac{\mathtt{LP}'}{B}\right)T\ \delta \\
&\leq (\mathtt{LP}'/B+1)\cdot T\cdot \mathtt{rad}_t(dK). \\
\mathtt{LP}'-f(\delta) &= T\ r' - T\ \min(r',r') \\
&\leq T\cdot \mathtt{rad}_t(dK) \\
\mathtt{LP}'-\mathtt{LP}'' &= (\mathtt{LP}'-f(\delta)) + (f(\delta)-\mathtt{LP}'') \\
&\leq (\mathtt{LP}'/B+2)\cdot T\cdot \mathtt{rad}_t(dK).
\end{aligned}
$$

Subcase 2. Assume Equation (4.18) fails. Then $\mathtt{LP}' = B\ r'/c_j'$ for some resource j. Note that $c_i' \leq c_j'$ and $c_j'' \leq c_i''$ by the choice of i and j.

From these inequalities and Lemma 4.4.1 we obtain $c_i'' \leq c_j' + \mathtt{rad}_t(dK)$. There-

fore,

$$\begin{aligned}
B\frac{r''}{c_i''} &\geq B\,\frac{r' - \texttt{rad}_t(dK)}{c'_j + \texttt{rad}_t(dK)} \qquad \text{(by Lemma 4.4.1)} \\
&\geq B\,\frac{r' - \texttt{rad}_t(dK)}{c'_j}\left(1 - \frac{\texttt{rad}_t(dK)}{c'_j}\right). \\
\texttt{LP}' - \texttt{LP}'' &= B\frac{r'}{c'_j} - B\frac{r''}{c''_i} \\
&\leq \left(\frac{B}{c'_j} + B\frac{r'}{(c''_j)^2}\right)\texttt{rad}_t(dK) \\
&\leq \left(T + \texttt{LP}'\frac{T}{B}\right)\texttt{rad}_t(dK) \\
&\leq (\texttt{LP}'/B + 1)\cdot T\cdot \texttt{rad}_t(dK). \qquad \square
\end{aligned}$$

4.8.2 The remainder of the proof after Lemma 4.4.4

We start with Corollary 4.4.5, which we restate here for convenience.

Corollary $\Phi_t \leq 2\Psi_t$, assuming that $B \geq 6\cdot T\cdot \texttt{rad}_t(dK)$.

Proof. Let $\gamma = \max_{P\in\mathcal{F}_\Pi,\mu\in\mathcal{I}_t} \texttt{LP}(P,\mu)$. Note that $\gamma \leq T$. Then from Lemma 4.4.4 we obtain:

$$\gamma - \texttt{OPT}_{\texttt{LP}} \leq 3(\tfrac{\gamma}{B} + 2)\cdot T\cdot \texttt{rad}_t(dK) \leq \tfrac{\gamma}{2} + 6\cdot T\cdot \texttt{rad}_t(dK). \tag{4.21}$$

Using (4.21) and Lemma 4.4.4 we get the desired bound:

$$\begin{aligned}
\Phi_t &\leq (\tfrac{\gamma}{B} + 2)\cdot T\cdot \texttt{rad}_t(dK) \\
&\leq \left(\frac{2\,\texttt{OPT}_{\texttt{LP}} + 12\cdot T\cdot \texttt{rad}_t(dK)}{B} + 2\right)\cdot T\cdot \texttt{rad}_t(dK) \\
&\leq \left(\frac{2\,\texttt{OPT}_{\texttt{LP}}}{B} + 4\right)\cdot T\cdot \texttt{rad}_t(dK) = 2\Psi_t. \qquad \square
\end{aligned}$$

In the remainder of this section, we prove the claims in Equation (4.13) one by one.

Corollary 4.8.1. $\texttt{REW}_t \geq \frac{t}{T}\,(\texttt{OPT}_{\texttt{LP}} - O(\Psi_t))$ *for each round* $t \leq \tau$.

Proof. From Lemma 4.4.6 we obtain

$$\begin{aligned} T\, r(P'_t, \mu) &\geq (1 - q_0)\, \texttt{LP}(P_t, \mu) \\ &\geq (1 - q_0)\,(\texttt{OPT}_{\texttt{LP}} - 12\Psi_t) \\ &\geq \texttt{OPT}_{\texttt{LP}} - 13\Psi_t. \end{aligned}$$

Summing up and taking average over rounds, we obtain:

$$T\,\overline{r}_t \geq \texttt{OPT}_{\texttt{LP}} - \tfrac{13}{t}\textstyle\sum_{s=1}^{t}\ \Psi_s \geq \texttt{OPT}_{\texttt{LP}} - O(\Psi_t).$$

By definition of clean execution, we obtain:

$$\texttt{REW}_t \geq t(\overline{r}_t - \texttt{rad}_t(\overline{r}_t)) \geq \tfrac{t}{T}(\texttt{OPT}_{\texttt{LP}} - O(\Psi_t)) \qquad \square$$

Corollary 4.8.2. $C_{t,i} \leq B/T + O(\texttt{rad}_t(dK))$ *for each round* $t \leq \tau$.

Proof. Let μ be the (actual) expected-outcomes tuple, and recall that P_t is LP-optimal for some expected-outcomes tuple $\mu' \in \mathbf{\Lambda}_t$. Then, by Lemma 4.4.1, it follows that $c_i(P_t, \mu) \leq c_i(P_t, \mu') + \texttt{rad}_t(dK)$. Furthermore since P_t is LP-optimal for μ' we have $c_i(P_t, \mu') \leq \frac{B}{T}$. Therefore:

$$\begin{aligned} c_i(P_t, \mu) &\leq \tfrac{B}{T} + \texttt{rad}_t(dK) \\ c_i(P'_t, \mu) &\leq (1 - q_0)\, c_i(P_t, \mu) + q_0 \\ &\leq \tfrac{B}{T} + O(\texttt{rad}_t(dK)). \end{aligned}$$

Now summing and taking average we obtain $\overline{c}_{t,i} \leq \frac{B}{T} + O(\texttt{rad}_t(dK))$. Using the definition of clean execution, it follows that

$$C_{t,i} \leq \overline{c}_{t,i} + \texttt{rad}_t(\overline{c}_{t,i}) \leq \tfrac{B}{T} + O(\texttt{rad}_t(dK)). \qquad \square$$

Lemma 4.8.3. $\texttt{REW}_T \geq \texttt{OPT}_{\texttt{LP}} - O(\Psi_T)$.

Proof. Either $\tau = T$ or some resource i gets exhausted, in which case (using Corollary 4.8.2)

$$\begin{aligned} \tau = \frac{B}{C_{\tau,i}} &\geq \frac{B}{\frac{B}{T} + \texttt{rad}_\tau(dK)} \\ \Rightarrow \tau\tfrac{B}{T} + \tau\texttt{rad}_\tau(dK) &\geq B \\ \Rightarrow \tau\tfrac{B}{T} + T\texttt{rad}_T(dK) &\geq B \\ \Rightarrow \tau &\geq T\left(1 - \tfrac{T}{B}\,\texttt{rad}_T(dK)\right). \end{aligned} \tag{4.22}$$

Using this lower bound and Corollary 4.8.1, we obtain the desired bound on the total revenue $\texttt{REW}_T$.

$$\begin{aligned} \texttt{REW}_T = \texttt{REW}_\tau &\geq \frac{\tau}{T}\left(\,\texttt{OPT}_{\texttt{LP}} - O(\Psi_\tau)\,\right) \\ &\geq \texttt{OPT}_{\texttt{LP}}(1 - \tfrac{T}{B}\,\texttt{rad}_T(dK)) - \frac{O(\tau\,\Psi_\tau)}{T} \\ &\geq \texttt{OPT}_{\texttt{LP}} - \Psi_T - \frac{O(\tau\,\Psi_\tau)}{T}. \end{aligned}$$

In the above, the first inequality holds by Corollary 4.8.1, the second by Equation (4.22), and the third by definition of Ψ_T.

Finally, we note that $\tau\,\Psi_\tau$ is an increasing function of τ, and substitute $\tau\Psi_\tau \leq T\Psi_T$. □

We complete the proof of Theorem 1.5.4 as follows. Re-writing Lemma 4.8.3 as $\texttt{REW}_T \geq f(\texttt{OPT}_{\texttt{LP}})$, for an appropriate function $f()$, note that $\texttt{REW}_T \geq f(\texttt{OPT})$ because function $f()$ is increasing.

CHAPTER 5

DYNAMIC PROCUREMENT

In this chapter we consider the special case of dynamic procurement and its generalizations. In section 5.1 we consider symmetric submodular procurement and propose a novel random walk based mechanism which achieves a constant factor approximation. Then in section 5.2 we consider the general submodular procurement and propose a guessing based mechanism which is a $O(\log n)$ approximation.

5.1 Posted Price Mechanisms for Unknown Distributions

In this section we assume that the utility function f is a a symmetric submodular function, and we design a $O(1)$-competitive posted price mechanism for the i.i.d. model. Our focus throughout most of this section is on the special case $f(S) = |S|$. In this special case, our goal can be restated as follows: we must show how a buyer with a budget of B can purchase approximately as many services as possible using a sequence of take-it-or-leave-it offers to sellers with i.i.d. costs, without knowing the distribution of costs in advance. Later, in Subsection 5.1.2, we show how this result for the function $f(S) = |S|$ easily implies a constant-factor approximation for general symmetric submodular f. The case of non-symmetric submodular functions is deferred to Sections 5.2 and 5.3. We emphasize that in this section we target only a constant-factor approximation, and we do not optimize for this constant.

It is instructive to compare our setting to an online single-item posted-price auction with i.i.d. bidders. For this problem [6] shows a $\Omega(\frac{\log h}{\log \log h})$ lower bound.

Here h is the ratio between maximum possible value and minimum possible value. Our result illustrates that the hardness of posted pricing in the i.i.d. model with an unknown distribution does not extend to budgeted procurement with a symmetric submodular utility function. To achieve a constant-approximation, we need to develop a novel algorithm that is quite different from existing algorithms in the literature on online posted pricing. Specifically, rather than reducing to a multi-armed bandit problem and using a general-purpose bandit algorithm (which is inapplicable in our setting because of the budget constraint) we design an algorithm that is quite different from bandit algorithms: in effect, it emulates a stochastic search for the optimal price using a multiplicative-increase multiplicative-decrease rule, with a simple sample-and-average policy guiding the multiplicative updates. We show, via a somewhat intricate analysis, that the stochastic search inflicts only a constant-factor loss relative to a buyer with foreknowledge of the optimal price. In effect, this involves showing that the algorithm has such a strong bias toward the optimal price that (with constant probability) a constant fraction of all the prices it offers are at or near the optimal price.

5.1.1 Algorithm for the Special Case $f(S) = |S|$

Before describing the details of the algorithm and its analysis, it will be useful to outline the main ideas. To begin with, consider what would happen if the algorithm were to offer a fixed price p to every agent. The budget constraint ensures that it cannot procure more than B/p services at this price, while the limitation on the number of agents ensures it cannot procure more than $n \cdot s(p)$ services in expectation, where $s(p)$ is the probability of a random agent accept-

ing an offer at price p. It is approximately optimal to offer a price that balances these two terms, i.e. a price p such that $B/p = \Theta(n \cdot s(p))$. Lemma 5.1.1 justifies this intuition.

Our algorithm descretizes the price range into a geometric progression $p_1, \ldots, p_m$ and the main loop of the algorithm emulates a Markov chain whose state set is the set $\{p_1, \ldots, p_m\}$. As explained in the preceding paragraph, states p that satisfy $B/p = \Theta(n \cdot s(p))$ play a special role in the analysis, and below we define a set of *good states* consisting of either one state or two consecutive states satisfying this relation.

The algorithm's goal is to search for a good state and then remain in a good state as often as possible thereafter. Since a good state is one in which $s(p) \cdot \frac{np}{B} = \Theta(1)$, we can test if a state p is good by making $\Theta\left(\frac{np}{B}\right)$ offers at price p and seeing if $\Theta(1)$ of them are accepted. This test is implemented in a subroutine denoted by TEST(i) in the algorithm description below. If the test produces significantly fewer than the anticipated number of successes, we know that the price is too low and we move to the next price in the geometric progression. Similarly, if the test produces significantly more than the anticipated number of successes, we know that the price is too high and we move to the preceding price in the geometric progression.

The foregoing discussion motivates Lemma 5.1.2 below, in which we show that the algorithm has a strong bias to drift toward the set of good states. Applying standard techniques from the analysis of random walks, this means that after finding a good state, it makes only very brief excursions away from the set of good states: each excursion occupies only a constant number of TEST($\cdot$) phases in expectation, and the distribution of the excursion lengths has an ex-

ponential tail. Lemma 5.1.4 capitalizes on this fact to prove bounds on the expected number of offers made and the expected amount of money spent during any such excursion. The bounds state that for each of these two resources (offers and money) the expected amount consumed during an excursion is $O(1)$ times the expected amount consumed during a single good phase (i.e., a TEST($\cdot$) phase carried out in a good state). The initialization phase, before the first time that the algorithm reaches a good state, can also be treated as an excursion for present purposes; see Corollary 5.1.5 below. The theorem that our algorithm is constant-competitive (Theorem 5.1.6) now follows by combining these observations: the analysis of random-walk excursions ensures that (in expectation) a constant fraction of our resources are consumed in good states, and the definition of good states ensures that resources consumed in good states are converted into accepted offers at the (approximately) optimal rate.

We present our algorithm in pseudo-code in Algorithm 6. We also explain our algorithm in plain english below.

The algorithm uses states $1, 2, \ldots, m$ with associated prices $p_1, p_2, \ldots, p_m$ that form a geometric progression with common ratio r. The progression starts at B/n, so that $p_1 = B/n$, $B/(z \cdot r) < p_m \leq B/z$, and $p_{i+1} = r \cdot p_i$ for $0 \leq i < m$. The main loop of the algorithm is a loop that, in state i, runs a subroutine TEST(i) whose output is an element of $\{i-1, i, i+1\}$. The output of TEST(i) becomes the new state.

Subroutine TEST(i) operates as follows. It offers price $p = p_i$ to anp/B bidders, for some constant a. If the number of accepted offers is ever greater than $a(1+\delta)$ it quits the subroutine immediately and outputs state $i-1$. If the number of successes is less than $a(1-\delta)$, then it outputs state $i+1$. Otherwise it outputs

Algorithm 6: RAND_WALK

$\delta = 1/10$, $r = 2$, $a = 4000$, $z = 300000$.

With probability 1/2:

Consider states $p_1, p_2, \ldots, p_m$ where $p_1 = \frac{B}{n}$, $p_{i+1} = r \cdot p_i$ for $1 \le i < m$, and $\frac{B}{rz} < p_m \le \frac{B}{z}$.

Initialize state $i = m$.

TEST(i): Set price $p = p_i$.

1. Offer price p to the next $\frac{anp}{B}$ sellers, or until the sellers or budget runs out.

   In case more than $a(1+\delta)$ offers are accepted, then stop offering and move to step 2.

2. Let t be the number of sellers who accept.

   i. If $a(1-\delta) \le t \le a(1+\delta)$ then go to TEST(i).

   ii. If $t > a(1+\delta)$ then update i to $i-1$ and go to TEST($i-1$).

   iii. If $t < a(1-\delta)$ then update i to $i+1$ and go to TEST($i+1$).

With probability 1/2:

Allocate to the first agent with cost $c_i \le B$

state i.

Analysis of the algorithm. Let ℓ be the index such that $B/p_{\ell-1} \ge n \cdot s(p_{\ell-1})$ and $B/p_\ell < n \cdot s(p_\ell)$ where $s(p)$ is the probability that a seller sells while offering at price p. (Note that B/p is a decreasing function of p whereas $s(p)$ is non-decreasing, hence p_ℓ is undefined if and only if $B/p_m \ge n \cdot s(p_m)$. But in such a case the second half of our algorithm — which with probability 1/2 sells to a

single player — is a $4rz$ approximation according to Lemma 5.1.1.)

Lemma 5.1.1. $\mathbb{E}(|OPT|) \leq 2r \cdot B/p_\ell$

Proof. Consider any specific realization of the random variables representing the costs of each seller. Define

$$\begin{aligned} O_1 &= \{i | c_i \leq p_{\ell-1} \text{ and } i \in OPT\} \\ O_2 &= \{i | c_i > p_{\ell-1} \text{ and } i \in OPT\} \\ O_3 &= \{i | c_i \leq p_{\ell-1}\}. \end{aligned}$$

We obtain our upper bound on $\mathbb{E}(|OPT|)$ by comparison with the expected cardinalities of O_1, O_2, O_3 as follows.

$$\begin{aligned} \mathbb{E}(|OPT|) &= \mathbb{E}(|O_1| + |O_2|) \\ &\leq \mathbb{E}(|O_3|) + \mathbb{E}(|O_2|) \\ &= ns(p_{\ell-1}) + \mathbb{E}(|O_2|) \\ &\leq ns(p_{\ell-1}) + \frac{B}{p_{\ell-1}} \leq \frac{B}{p_{\ell-1}} + \frac{B}{p_{\ell-1}} = 2r\frac{B}{p_\ell} \end{aligned}$$

□

Define the set of *good states* to be $GS = \{p_i | i \leq \ell \text{ and } s(p_i) \geq \frac{B(1-2\delta)}{np_i}\}$. Note that $p_\ell \in GS \subseteq \{p_{\ell-1}, p_\ell\}$. In a good state p_i, the expected number of accepted offers during TEST(i) is

$$\frac{anp_i s(p_i)}{B} \geq (1 - 2\delta)a.$$

This partially justifies the term "good states", since the algorithm is designed to find a state in which the expected number of accepted offers equals a.

Lemma 5.1.2. *If $p = p_i$ satisfies $p < p', \forall p' \in GS$ then TEST(i) outputs $i + 1$ with probability at least $1 - (1 - 2\delta)/(\delta^2 a)$. If $p > p', \forall p' \in GS$ then TEST(i) outputs $i - 1$ with probability at least $1 - (1 - 2\delta)/(\delta^2 a)$.*

Proof. We will use the fact that the variance of a Bernoulli random variable is bounded above by its expectation; consequently the same property holds for sums of independent Bernoulli random variables. By an application of Chebyshev's Inequality, then, if the expectation of a sum of Bernoulli random variables is y, then the probability that the sum differs from its expectation by more than w is at most y/w^2.

If $s(p) < \frac{(1-2\delta)B}{np}$ then the expected number of successes in TEST(i) is less than $\frac{anp}{B} \cdot \frac{(1-2\delta)B}{np} = (1-2\delta)a$. Let $y \in [0, (1-2\delta)a]$ be the expected number of successes. The probability that the number of successes is greater than or equal to $a(1-\delta)$ is at most $\frac{y}{(a(1-\delta)-y)^2} \leq \frac{(1-2\delta)a}{\delta^2 a^2} = \frac{1-2\delta}{\delta^2 a}$.

If $s(p) \geq s(p_l) > \frac{B}{np_l} \geq \frac{rB}{np} > \frac{B}{(1-2\delta)np}$ then the expected number of successes in TEST(i) is greater than $\frac{anp}{B} \cdot \frac{B}{(1-2\delta)np} = (1-2\delta)^{-1}a$. The probability that the number of successes is less than or equal to $a(1+\delta)$ is at most $\frac{a/(1-2\delta)}{(\delta+2\delta^2)^2 a^2/(1-2\delta)^2} \leq \frac{1-2\delta}{\delta^2 a}$. □

Informally, Lemma 5.1.2 says that when the current price is far from the good states, the random walk is biased to drift in the direction of the good states. Analyzing this biased random walk is the crux of our proof. To simplify the analysis, we couple the algorithm's random walk with a simpler random walk in which the bias to drift toward the good states is exactly equal to the lower bound asserted in Lemma 5.1.2. Denote this bias by $\beta = 1 - (1-2\delta)/(\delta^2 a)$. The algorithm's Markov chain $\mathcal{M}_1$ is defined to have states $1, 2, \ldots, m$, and the transition probability from state i to state $j \in \{i-1, i, i+1\}$ is equal to the probability that an execution of TEST(i) finishes by calling TEST(j). (Conditional on having a nonzero budget and number of sellers remaining at the end of TEST(i).) We compare this Markov chain to another one, $\mathcal{M}_2$, defined on the set of nonnegative integers. State 0 is an absorbing state of $\mathcal{M}_2$ and in every other state

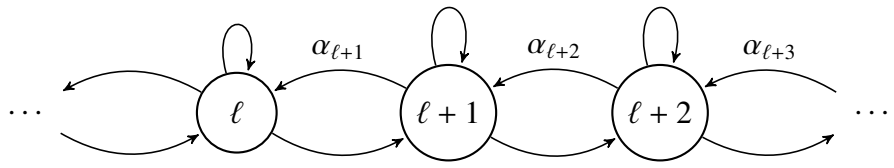

Figure 5.1: Actual Markov chain with $\alpha_i \geq \beta$ for $i \geq l+1$

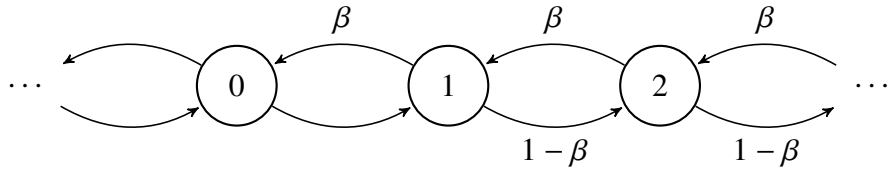

Figure 5.2: New Markov chain

$i > 0$, the transition probabilities to states $i-1$ and $i+1$ are β and $1-\beta$, respectively. The two Markov chains are depicted in Figures 5.1 and 5.2.

For $1 \leq i \leq m$, let $\Delta(i)$ denote the distance from state i to the set of good states, i.e.

$$\Delta(i) = \min_{j \in GS} |i - j|.$$

The following lemma, stated informally, says the Markov chain $\mathcal{M}_2$ starting from state $\Delta(i)$ provides an upper bound on the stochastic process describing the distance from GS when one starts $\mathcal{M}_1$ in state i and runs it until it reaches GS.

Lemma 5.1.3. *Let i be any state of $\mathcal{M}_1$. There is a coupling of $\mathcal{M}_1$ and $\mathcal{M}_2$ such that $\mathcal{M}_1$ starts in state $i_0 = i$, $\mathcal{M}_2$ starts in state $k_0 = \Delta(i)$, and at any time $t \geq 0$ if the first Markov chain's state sequence $i_0, i_1, \ldots, i_{t-1}$ does not include any element of GS, then the current pair of states (i_t, k_t) satisfies $k_t \geq \Delta(i_t)$.*

Proof. The coupling is easy to describe. When the two Markov chains are in states (i, k), respectively, if $i \in GS$ we couple them arbitrarily, e.g. by updating i

and k independently using the transition probability of their respective Markov chains. If $i \notin GS$ let i' denote the neighboring state that is closer to GS than i: $i' = i + 1$ if $i < j$ for all $j \in GS$, and $i' = i - 1$ if $i > j$ for all $j \in GS$. Let α_i denote the transition probability from i to i' in $\mathcal{M}_1$, and let σ_i denote the transition probability from i to itself in $\mathcal{M}_1$. Our coupling in state (i, k) works as follows. With probability β we update i to i' and k to $k - 1$. With the remaining $1 - \beta$ probability, we update k to $k + 1$ and update i as follows: it transitions to i' with probability $\alpha_i - \beta$, to i with probability σ_i, and to $2i - i'$ (the other neighboring state) with probability $1 - \alpha_i - \sigma_i$. It is immediate from our definition of the coupling that if (i_{t-1}, k_{t-1}) and (i_t, k_t) are two consecutive state pairs such that $i_{t-1} \notin GS$, then

$$\Delta(i_t) - \Delta(i_{t-1}) \leq k_t - k_{t-1}. \tag{5.1}$$

The lemma follows by summing Equation (5.1) over all time steps preceding t. □

Lemma 5.1.4. *Consider a sequence of phases in our algorithm's execution beginning with TEST(i) where $i \notin GS$ and ending immediately before the first subsequent instance of TEST(j) such that $j \in GS$. In total during this sequence of phases, the algorithm makes no more than $(anp_\ell/B)\phi(\Delta(i))$ offers in expectation and spends no more than $a(1 + \delta)p_\ell\phi(\Delta(i))$ of its budget in expectation, where $\phi(\cdot)$ is the function*

$$\phi(k) = \frac{r}{r - \beta - (1 - \beta)r^2} \cdot (r^k - 1). \tag{5.2}$$

Proof. Let ℓ' denote the minimum element of GS (either ℓ or $\ell - 1$). For a given value Δ, there are at most two states i such that $\Delta(i) = \Delta$, namely $i = \ell + \Delta$ and $i = \ell' - \Delta$. In state $\ell + \Delta$, the algorithm makes $anp_i/B = anp_\ell r^\Delta/B$ offers in expectation, and it spends at most $a(1 + \delta)p_i = a(1 + \delta)p_\ell r^\Delta$ because subroutine TEST(i) stops making offers after $a(1 + \delta)$ offers have been accepted. In state

$\ell' - \Delta$ the algorithm makes $anp_{\ell'} r^{-\Delta}/B$ offers in expectation, and it spends at most $a(1+\delta)p_{\ell'} r^{-\Delta}$. Thus, for a given value of Δ, the expected number of offers made and the expected amount spent in state $\ell + \Delta$ are both greater than their counterparts in state $\ell' - \Delta$. Accordingly, it suffices to prove the lemma for $i = \ell + \Delta$.

To do so, we use the coupling provided by Lemma 5.1.3. Our algorithm proceeds through a sequence of states $i_0, i_1, \ldots, i_T$ such that i_T is the first state in the sequence that belongs to GS. Meanwhile Markov chain $\mathcal{M}_2$ proceeds through a sequence of states $k_0, k_1, \ldots, k_T$ such that $k_t \geq \Delta(i_t)$ for all $t = 0, \ldots, T$. By the foregoing discussion, the expected number of offers made by our algorithm and the expected amount spent are respectively bounded above by

$$\begin{aligned}
\mathbb{E}[\text{offers made}] &\leq \sum_{t=0}^{T-1} anp_\ell r^{k_t}/B = \tfrac{anp_\ell}{B} \sum_{t=0}^{T-1} r^{k_t} \\
\mathbb{E}[\text{amount spent}] &\leq \sum_{t=0}^{T-1} a(1+\delta)p_\ell r^{k_t} = a(1+\delta) \sum_{t=0}^{T-1} r^{k_t}.
\end{aligned}$$

Thus, to complete the lemma, it suffices to bound $\mathbb{E}\left[\sum_{t=0}^{T-1} r^{k_t}\right]$. We do so by introducing another stopping time τ, defined to be the earliest t such that $k_t = 0$. Note that $\tau \geq T$, so $\sum_{t=0}^{\tau-1} r^{k_t} \geq \sum_{t=0}^{T-1} r^{k_t}$. Define

$$\phi(k) = \mathbb{E}\left[\sum_{t=0}^{\tau-1} r^{k_t} \,\middle|\, k_0 = k\right].$$

The function ϕ satisfies the linear recurrence

$$\phi(k) = r^k + \beta\phi(k-1) + (1-\beta)\phi(k+1)$$

for $k > 0$, with the initial condition $\phi(0) = 0$. Solving the recurrence we find that equation Equation (5.2) in the statement of the lemma specifies the solution. □

In the sequel, let $c_1 = \frac{r}{r-\beta-(1-\beta)r^2}$.

Corollary 5.1.5. *Consider the sequence of phases beginning with the algorithm's initialization and ending with the first subsequent instance of TEST(j) such that $j \in GS$. In total during this sequence of phases, the algorithm makes no more than $(ac_1/z) \cdot n$ offers in expectation and spends no more than $(a(1+\delta)c_1/z) \cdot B$ of its budget in expectation.*

Proof. The algorithm begins in phase m. Using the fact that $r^{m-1} \leq n/z$, this implies the bound $\phi(\Delta(m)) = c_1(r^{m-\ell} - 1) < (c_1 r^{1-\ell})n/z$. By Lemma 5.1.4, the expected number of offers made before reaching a good state is at most

$$\frac{anp_\ell}{B}(c_1 r^{1-\ell})(n/z) = \frac{anr^{\ell-1}(B/n)}{B}(c_1 r^{1-\ell})(n/z) = \left(\frac{ac_1}{z}\right)n.$$

The bound in the expected amount spent follows by an analogous calculation:

$$a(1+\delta)p_\ell(c_1 r^{1-\ell})(n/z) = a(1+\delta)r^{\ell-1}(B/n)c_1 r^{1-\ell}n/z = (a(1+\delta)c_1/z) \cdot B.$$

□

Theorem 5.1.6. *The algorithm is constant-competitive. In fact, with probability at least $\frac{1}{2}$, the number of offers accepted is at least $c \cdot (B/p_\ell)$, where c is an absolute constant.*

Proof. We distinguish two cases. If $B/p_\ell \leq 1.2 \times 10^5$ then with probability $1/2$, we expend our full budget on a single agent. When this happens, the number of offers accepted equals 1, which is at least $(2.4 \times 10^{-5})(B/p_\ell)$.

If $B/p_\ell \geq 1.2 \times 10^4$ then we will show that with constant probability, the algorithm executes enough calls to TEST(i) for states $i \in GS$ that it succeeds in getting $c \cdot (B/p_\ell)$ offers accepted. We will define five bad events $E_1, \ldots, E_5$, each having probability at most 0.1, such that the number of offers accepted is greater than $c \cdot (B/p_\ell)$ whenever none of the events $E_1, \ldots, E_5$ occurs. Corollary 5.1.5 presents

bounds on the expected number of offers made, and the expected amount spent, during the "startup stage" before the first time the algorithm reaches a good state. Let E_1 (resp. E_2) denote the event that the actual number of offers made (resp. amount spent) in the startup stage exceeds the bound given in Corollary 5.1.5 by a factor of more than 10. By Markov's inequality, each of E_1, E_2 has probability bounded by 0.1.

Imagine running the algorithm for an infinite number of steps, disregarding the fact that it eventually spends more than its budget and makes more than its allotted n offers. Define a *good phase* to be an execution of TEST(i) such that $i \in GS$. Define an *excursion* to be the (possibly empty) set of offers made between two consecutive good phases. Let

$$x = \frac{B}{p_l} \min \left\{ \frac{1 - \frac{ac_1}{z}}{a(1 + 10c_1(r-1))}, \frac{1 - \frac{ac_1(1+\delta)}{z}}{a(1+\delta)(1 + 10c_1(r-1))} \right\}$$

and consider the first x good phases along with the x excursions that occur immediately after each of them (Note, by our assumption that $B/p_l \geq 1.2 \times 10^5$ we have $x \geq 1$). How many offers, in expectation, are made during these x phases and excursions? The expected number of offers during the good phases is bounded above by $(anp_\ell/B) \cdot x$. According to Lemma 5.1.4 the expected number of offers during the excursions is bounded above by $(anp_\ell/B)c_1(r-1)$ and the expected amount spent during the excursions is bounded above by $a(1+\delta)p_\ell c_1(r-1)$. Let E_3 (resp. E_4) denote the event that the actual number of offers (resp. amount spent) made during the excursions does not exceed this bound by a factor of more than 10. Once again, by Markov's inequality, each of E_3, E_4 has probability bounded by 0.1.

Our choice of the parameters a, r, z, δ has been designed to ensure that, assuming events E_1, E_2, E_3, E_4 do not occur, the combined number of offers made

until the end of the x^{th} good phase is less than n and the combined amount spent until this time is less than B. Let N' be the number of offers made and let B' be the budget spent before x good phases assuming events E_1, E_2, E_3, E_4 do not occur.

$$\begin{aligned} N' = &\text{ Offers before first good phase+in good phases+in recursions} \\ \leq &\quad \tfrac{10ac_1}{z}n + \tfrac{anp_l}{B}x + \tfrac{10anp_lc_1(r-1)}{B}x \\ \leq &\quad \tfrac{10ac_1}{z}n + (\tfrac{anp_l}{B} + \tfrac{10anp_lc_1(r-1)}{B})\tfrac{B}{p_l}\tfrac{1-\frac{ac_1}{z}}{a(1+10c_1(r-1))} \\ \leq &\quad n \end{aligned}$$

$$\begin{aligned} B' = &\text{ Budget spent before first good phase+in good phases+in recursions} \\ \leq &\quad \tfrac{10ac_1(1+\delta)}{z}B + ap_l(1+\delta)x + 10ap_l(1+\delta)c_1(r-1)x \\ \leq &\quad \tfrac{10ac_1(1+\delta)}{z}B + (ap_l(1+\delta) + 10ap_l(1+\delta)c_1(r-1))\tfrac{B}{p_l}\tfrac{1-\frac{ac_1(1+\delta)}{z}}{a(1+\delta)(1+10c_1(r-1))} \\ \leq &\quad B \end{aligned}$$

Thus, all of the offers made during the first x good phases actually took place during the algorithm's execution, i.e. before the budget was expended or the maximum number of offers was reached. In any good phase, the expected number of offers accepted is at least $a(1-2\delta)$. Let E_5 denote the event that fewer than $a(1-3\delta)x$ offers are accepted in all of the first x good phases combined. Applying Chebyshev's Inequality to this sum of independent Bernoulli random variables as in the proof of Lemma 5.1.2 implies that $\Pr(E_5) \leq \frac{1-2\delta}{\delta^2 a} \leq 0.1$, again by our choice of a, r, z, δ.

By the union bound, the probability that none of $E_1, \ldots, E_5$ occur is at least $\frac{1}{2}$, and when this happens at least $a(1-3\delta)x$ offers are accepted. The theorem follows, because $x = \Omega(B/p_\ell)$. □

5.1.2 Extension to Symmetric Submodular Functions

When f is a symmetric submodular function, it means that there exists a non-decreasing concave function g such that $f(S) = g(|S|)$ for all sets S. It turns out that the argument from Section 5.1.1 carries through to this case with very few modifications. As an upper bound on $\mathbb{E}[f(OPT)]$, we use the following lemma which generalizes Lemma 5.1.1.

Lemma 5.1.7. $\mathbb{E}[f(OPT)] \leq 2r \cdot g(B/p_\ell)$

Proof. Consider any specific realization of the random variables representing the costs of each seller. Define

$$\begin{aligned} O_1 &= \{i | c_i \leq p_{\ell-1} \text{ and } i \in OPT\} \\ O_2 &= \{i | c_i > p_{\ell-1} \text{ and } i \in OPT\} \\ O_3 &= \{i | c_i \leq p_{\ell-1}\}. \end{aligned}$$

We obtain our upper bound on $\mathbb{E}[f(OPT)]$ via the following manipulation, whose first, second, and last lines follow from the fact that g is a concave function satisfying $g(0) = 0$.

$$\begin{aligned} \mathbb{E}[f(OPT)] &= \mathbb{E}[g(|OPT|)] \leq \mathbb{E}[g(|O_1|) + g(|O_2|)] \\ &\leq g(\mathbb{E}(|O_3|)) + g(\mathbb{E}(|O_2|)) \\ &= g(ns(p_{\ell-1})) + g(\mathbb{E}(|O_2|)) \\ &\leq g(ns(p_{\ell-1})) + g\left(\tfrac{B}{p_{\ell-1}}\right) \leq 2g\left(\tfrac{B}{p_{\ell-1}}\right) \leq 2r \cdot g\left(\tfrac{B}{p_\ell}\right) \end{aligned}$$

□

By Theorem 5.1.6, with probability at least $\frac{1}{2}$ the number of accepted offers is at least $c_5 \cdot (B/p_\ell)$. Hence, letting S denote the set of accepted offers,

$$\mathbb{E}[f(S)] \geq \tfrac{1}{2} c_5 \cdot g\left(\tfrac{B}{p_\ell}\right) \geq \tfrac{c_5}{4r} \cdot \mathbb{E}[f(OPT)],$$

where the last inequality follows from Lemma 5.1.7. This completes the proof that our algorithm is constant-competitive whenever f is a nondecreasing symmetric submodular function.

5.2 An $O(\log n)$ Posted Price Mechanism for Submodular Markets

In this section we present a posted price mechanism which is $O(\log n)$-competitive for any nondecreasing submodular utility function, in the secretary model. A function $f : 2^{[n]} \to \mathbb{R}_+$ is submodular if for any subsets S, T s.t. $S \subseteq T$ we have $f(S \cup \{a\}) - f(S) \geq f(T \cup \{a\}) - f(T)$. The function is called *nondecreasing* if $S \subseteq T$ implies $f(S) \leq f(T)$.

Theorem 5.2.1. *For any nondecreasing submodular procurement market there is a randomized posted price budget feasible mechanism which is universally truthful and is $O(\log n)$-competitive.*

In proof, we present the following mechanism, and show it respects the mentioned properties. In its essence, the mechanism is quite simple: it samples approximately half of the agents, rejects them and finds the agent with the highest

value in the sample. It then uses this value to guess a threshold price that can be used to decide on the allocation of the remaining agents.

Algorithm 7: GUESS_POST

With probability 1/2 do ALG1:

ALG1

1. Choose $\tau \in [0, n]$ with $Pr[\tau = i] = \binom{n}{i}/2^n$.
2. Offer $p = 0$ to the first τ agents that arrive and let $v' = \max_{\{a_i : i \leq \tau\}} f(a_i)$.
3. Choose i u.a.r from $\{0, 1, 2, \ldots, \lceil \log n \rceil\}$, let $t = 2^i v'$ and $B' = B$
4. For each agent $a \in \mathcal{N} \setminus \{a_1, \ldots, a_\tau\}$:
   a. Offer the agent $p = \frac{B}{t} \cdot (f(S \cup \{a\}) - f(S))$ if $B' - p \geq 0$
   b. If a accepts, add her to S and set $B' = B' - p$.

Otherwise:

ALG2

Run Dynkin's algorithm and offer B to the winner

The mechanism runs Dynkin's algorithm with probability 1/2 to allocate to an agent with a sufficiently high value [24]. That is, with probability 1/2 the mechanism samples the first n/e agents and then, from the remaining $(1 - 1/e)n$ agents, allocates to the first agent a' for which $f(a') \geq \text{argmax}_{i \in \{1, \ldots, n/e\}} f(a_i)$. In expectation over the arrival order of the agents this guarantees that $f(a') \geq (1/e) \cdot \text{argmax}_{a \in \mathcal{N}} f(a)$.

Since the mechanism is a randomization over two posted price mechanisms, it is truthful and individually rational. Budget feasibility is implied from the condition in (4a) which verifies that the remaining budget B' is greater than the payment. Throughout the rest of this paper we will use $OPT(\mathcal{N}')$ to denote the

optimal value over the set of agents $\mathcal{N}'$, $\mathcal{N}_1$ to denote the set of agents who are in the sample and $\mathcal{N}_2 = \mathcal{N} \setminus \mathcal{N}_1$. Let $S^* = OPT(\mathcal{N})$, $S_1^* = S^* \cap \mathcal{N}_1$ and $S_2^* = S^* \cap \mathcal{N}_2^*$.

Let $v^* = max\{f(a), a \in \mathcal{N}\}$. Consider the case when $f(v^*) \geq f(OPT)/1024$. In this case the algorithm runs Dynkin's algorithm with probability $1/2$ and gets an approximation ratio of $2048e$. So for the rest of the section we will assume that $f(a) < f(OPT)/1024, \forall a \in \mathcal{N}$. To prove the competitive ratio of the mechanism, we will use the following lemma.

Lemma 5.2.2. *When* $f(a) \leq \frac{f(OPT)}{1024}, \forall a \in \mathcal{N}$ *then* $\min\{f(S_1^*), f(S_2^*)\} \geq \frac{OPT(\mathcal{N})}{4}$ *with probability at least* $p \geq \frac{9}{10}$.

Proof. Let $S^* = \{a_1, \ldots, a_\ell\}$ be the optimal solution. Without loss of generality, assume that $a_1, \ldots, a_\ell$ are sorted according to decreasing marginal contributions, and let w_i denote the marginal contribution of a_i.

Since the agents are assumed to arrive in a uniformly random order, and τ is chosen with a specific probability distribution we have that the sampled set $\mathcal{N}_1$ is a uniformly random set of $\mathcal{N}$. Hence each agent is in $\mathcal{N}_1$ with probability $1/2$ independently of other agents. Consider the random variables $X_1, \ldots, X_\ell$, s.t. X_i takes the value w_i with if agent i belongs to $\mathcal{N}_1$ and 0 otherwise. Since $\mathcal{N}_1$ is a uniformly random set this implies that X_i takes value w_i with probability $1/2$ and 0 with probability $1/2$. Such a trick of choosing set $\mathcal{N}_1$ so that X_i's are independent was first introduced by [37]. Let $X = \sum_{i=1}^{\ell} X_i$ and $\bar{X} = f(S^*) - X$. Observe that $f(S_1^*) \geq X$, and $f(S_2^*) \geq \bar{X}$ by submodularity. To show the desired properties of X and $\bar{X}$ we will use the Chernoff bound stated in lemma 2.2.3.

Since our assumption that $max_i w_i \leq \frac{f(S^*)}{1024}$ and that $\mu = f(S^*)/2$ the above bound implies that:

$$Pr\Big[\bar{X} \leq \frac{f(S^*)}{4}\Big] = Pr\Big[X \geq \frac{3f(S^*)}{4}\Big] \leq 1/20 \tag{5.3}$$

as well as:

$$Pr\Big[\bar{X} \geq \frac{3f(S^*)}{4}\Big] = Pr\Big[X \leq \frac{f(S^*)}{4}\Big] \leq 1/20 \tag{5.4}$$

Let $W_1 = \sum_{i \in S_1^*} w_i$ and $W_2 = \sum_{i \in S_2^*} w_i$. By union bound, the above inequalities imply that with probability at least $p \geq 1 - (1/20 + 1/20) = 9/10$ we have that:

$$W_2/3 \leq W_1 \leq 3W_2 \tag{5.5}$$

Since $OPT = W_1 + W_2$, this implies that both W_1 and W_2 are greater than $OPT/4$ with probability p. □

Lemma 5.2.3. *When $f(a) \leq \frac{f(OPT)}{1024} \leq \frac{4f(S_2^*)}{1024}$, for any realization of t where $\frac{f(S_2^*)}{64} \leq t \leq \frac{f(S_2^*)}{2}$ ALG1 is a 128-approximation to $f(S_2^*)$.*

Proof. Let S be the set of all agents that received the mechanism's offer. Since agents are rational and the mechanism is incentive compatible, an agent a_i which rejects the mechanism's offer p_i implies that $c_i > p_i$. Note that in order to be included in S, an agent a_i needs to respect: $c_i \leq p_i \leq B'$, where B' is the remaining budget at stage i in which a_i appears. First, consider the case where every agent $a_i \in S_2^* \setminus S$ rejects the mechanism's offer in ALG1, i.e. $c_i > p_i$. In this case we have:

$$\sum_{a \in S_2^* \setminus S} \Big(f(S \cup \{a\}) - f(S)\Big) = \sum_{a \in S_2^* \setminus S} \Big(\frac{f(S \cup a) - f(S)}{c_i}\Big) \cdot c_i < \sum_{a \in S_2^* \setminus S} \Big(\frac{t}{B}\Big) \cdot c_i \leq \frac{f(S_2^*)}{2B} \cdot B$$

where the first inequality is due to the fact that S_2^* is a feasible solution ($\sum_{a_i \in S_2^*} c_i \leq B$), and the second inequality is due to the definition of t. This inequality implies:

$$f(S_2^*) - f(S) \quad = \quad f((S \cup S_2^* \setminus S)) - f(S) \leq \sum_{a \in S_2^* \setminus S} \Big(f(S \cup a) - f(S) \Big) \leq \frac{f(S_2^*)}{2}$$

and we have that $f(S) > f(S_2^*) - f(S_2^*)/2 = f(S_2^*)/2$. Hence in this case it is a factor 2 approximation to $f(S_2^*)$.

Consider the second case where there is an agent $a_i \in S_2^* \setminus S$ for which $c_i < p_i$, this implies that at stage i, $B' < p_i$. Let a_i be the first agent in $S_2^* \setminus S$ which has this property. Now, there are two (sub)cases to consider. The first case is that $B' \leq B/2$, i.e. $\sum_{a_j \in S} p_j \geq B/2$:

$$\frac{B}{2} < \sum_{a_j \in S} p_j = \sum_{a_j \in S} \frac{B}{t} \Big(f(S_j \cup \{a_j\}) - f(S_j) \Big) \leq B \cdot \frac{64 f(S)}{f(S_2^*)} \tag{5.6}$$

which implies that $f(S)$ is a 128 approximation of $f(S_2^*)$. Finally, in the case where $B' < p_i$ and $B' > B/2$, we have:

$$\frac{B}{2} < B' < p_i = \frac{B}{t} \Big(f(S_j \cup \{a_i\}) - f(S_j) \Big) \leq \frac{B}{t} \cdot f(a_i) \leq \frac{64B}{f(S_2^*)} \cdot f(a_i) \leq \frac{64B}{256} \tag{5.7}$$

Here equation 5.7 is a contradiction. Hence the case that $B' > B/2$ never happens. □

There are two cases to show that the algorithm is $O(\log n)$ competitive ratio.

1. If $f(v^*) \geq f(OPT)/1024$ then we ALG2 is a $1024e$ approximation which is run with probability 1/2. In this case we get a $2048e$ approximation as remarked above.
2. If $f(a) \leq f(OPT)/1024, \forall a \in \mathcal{N}$, we consider the approximation ratio of ALG1 which is run with probability 1/2. With probability 1/2 v^* is included in $\mathcal{N}_1$(Event T_1) and with probability at least 9/10 we have that

$f(S_2^*) \geq f(OPT)/4$ by lemma 5.2.2(Event T_2). Hence by union bound both these events(Events T_1 and T_2) happen with probability at least $4/10 = 1 - (1/10 + 1/2)$. Independently of these events t such that $f(S_2^*)/64 \leq t \leq f(S_2^*)/2$ is chosen with probability $1/O(\log(n))$ which results in an approximation ratio of 128 by lemma 5.2.3. Hence the final approximation ratio is $\frac{1}{2}\frac{4}{10}\frac{1}{\log(n)}\frac{1}{4}\frac{1}{128} = \frac{1}{O(\log(n))}$.

This gives us Theorem 5.2.1 which is the main result of this section.

5.3 A Constant-Competitive Mechanism in the Bidding Model

In this section we present a bidding mechanism for nondecreasing submodular markets in the secretary model. As each agent arrives, the mechanism collects is bid and must make an irrevocable decision of whether or not the agent should be allocated and how much the agent should be rewarded. We will show the following theorem.

Theorem 5.3.1. *In the bidding model, for any nondecreasing submodular utility function, there is a universally truthful budget feasible mechanism which is $O(1)$-competitive.*

The result of the previous section showed that the gap between posted price mechanisms and bidding mechanisms in nondecreasing procurement markets in the secretary model is at most $O(\log n)$. The above theorem and theorem 5.1.6 which are a special cases of model considered in section 5.2, hint towards the possibility that the gap in theorem 5.2.1 can be reduced to $O(1)$.

Like the mechanism from the previous section, the mechanism here will also sample the agents and use a threshold value to decide on the allocation. In the process of estimating the threshold, the mechanism will compute an approximation of the optimal solution of the bids of the first half of the agents. A modification of the greedy algorithm that sorts agents according to their density – their marginal contribution normalized by their cost – achieves an approximation ratio of $e/(e-1)$ [33, 49] which is known to be optimal [25]. We use $A(\mathcal{N}_1)$ to denote the value of this algorithm computed over a subset of agents $\mathcal{N}_1 \subseteq \mathcal{N}$.

Algorithm 8: `Online_Bidding`

With probability 1/2 do ALG1:

ALG1

1. Choose $\tau \in [0, n]$ with $Pr[\tau = i] = \binom{n}{i}/2^n$.
2. Let $\mathcal{N}_1$ be the first τ agents that arrive, $\mathcal{N}_2 = \mathcal{N} \setminus \mathcal{N}_1$, and $B' = B$.
3. For each agent $a_i \in \mathcal{N}_1$:
   a. Add the a_i's bid to the sample;
   b. Reject a_i's bid
4. Compute $t = A(\mathcal{N}_1)/8$
5. For each agent $a_i \in \mathcal{N}_2$:
   If $c_i \leq p_i = \frac{B}{t} \cdot (f(S \cup \{a\}) - f(S))$ and $B' - p_i \geq 0$:
   add a_i to S, pay her p_i and set $B' = B - p_i$.

Otherwise:

ALG2

Run Dynkin's algorithm and pay the winner B.

The mechanism above is similar to the one from Section 5.2 though it does not guess a threshold at random, but rather leverages the access it has to agents'

bids to compute a "good" threshold value t from the sample. We will show that this threshold estimates OPT well, which allows for a good approximation. First though, we verify truthfulness.

Lemma 5.3.2. *The above mechanism is universally truthful.*

Proof. Fix an arrival sequence of the agents. An agent $a \in \mathcal{N}_1$ is always rejected by the mechanism, regardless of her bid, and she therefore cannot be better off by misreporting her cost. For each agent $a \in \mathcal{N}_2$ the threshold price it is offered is independent of her bid since agents have no control over their arrival order, and cannot control their marginal contributions.

Assume for purpose of contradiction that an agent a_i benefits from declaring a bid $b_i \neq c_i$. If the agent is allocated when declaring her true cost c_i, she receives the same price as long as her bid is below the threshold. Therefore she cannot benefit from declaring a cost which is higher or lower than her true cost and is below the threshold. Bidding above the threshold excludes the agent from the allocation and her utility in this case is 0, and she is worst off since the fact that she is allocated implies that her cost is below the threshold.

In case the agent is not allocated, she cannot benefit from bidding a higher cost or any cost above the threshold. In case the agent declares a cost which is below the threshold, she can get allocated. However, since her true cost is above the threshold, her utility would be negative, and she is better off declaring her true cost. □

Budget feasibility and individual rationality are maintained by enforcing the conditions in step (4) of the mechanism. Showing that the mechanism is indeed constant competitive would there imply the main result of this section.

Lemma 5.3.3. *The mechanism is $O(1)$-competitive.*

Proof. Once again we have two cases to argue about. The first case consider that $f(v^*) \geq f(OPT)/1024$. Then with probability 1/2 we are running dynkin's algorithm and getting in expectation $\frac{1}{e} max_{a \in \mathcal{N}} f(a)$. Hence this is a $2 \cdot e \cdot 1024 = 2048e$ approximation.

As in Lemma 5.2.2 let $S^* = \{a_1, \ldots, a_\ell\}$ be the optimal solution $OPT(\mathcal{N})$ and without loss of generality assume that $a_1, \ldots, a_\ell$ are sorted according to decreasing marginal contributions. Let w_i denote the marginal contribution of a_i, $S_1^* = S^* \cap \mathcal{N}_1, S_2^* = S^* \setminus S_1^*$ and $W_1 = \sum_{a_i \in S_1^*} w_i$ and $W_2 = \sum_{a_i \in S_2^*} w_i$.

Consider the second case where $\forall a \in \mathcal{N}, f(a) \leq f(OPT)/1024$. In such a case by lemma 5.2.2 with probability 9/10 we have that $\min\{f(S_1^*), f(S_2^*)\} \geq \frac{OPT(\mathcal{N})}{4}$. Given this event we will show that

$$\frac{f(S_2^*)}{64} \quad \leq \quad t \leq \frac{f(S_2^*)}{2}; \tag{5.8}$$

Recall that in Lemma 5.2.3, we showed that using a threshold with property (5.8) to allocate to agents in $\mathcal{N}_2$ if and only if the ratio between the threshold and their marginal contribution exceeds their cost to budget ratio (as described in step (4) of the mechanism), guarantees that in expectation the set of the allocated agents is a constant factor approximation of $f(S_2^*)$. Hence, showing property (5.8) above would imply that with a constant probability the value of the set of agents allocated by the mechanism is a constant factor approximation of $f(S_2^*)$. From Lemma 5.2.2 this implies the mechanism is constant competitive.

To compute the threshold t, in step (3) we apply the greedy algorithm for submodular maximization under a budget constraint on the sample $\mathcal{N}_1$. This

algorithm is guaranteed to provide at least a $(1 - 1/e)$ fraction of $OPT(\mathcal{N}_1)$. Due to the decreasing marginal utilities property of the submodular function, we have that $OPT(\mathcal{N}_1) \geq f(S_1^*) \geq W_1$ and that $OPT(\mathcal{N}_2) \geq f(S_2^*) \geq W_2$. From Lemma 5.2.2 we know that there is a constant probability for which $OPT(\mathcal{N})/4 \leq \min\{f(S_1^*), f(S_2^*)\}$. The threshold t can be bounded from above:

$$t = \frac{A(\mathcal{N}_1)}{8} \leq \frac{OPT(\mathcal{N}_1)}{8} \leq \frac{OPT(\mathcal{N})}{8} \leq \frac{f(S_2^*)}{2} \tag{5.9}$$

To bound t from below we use $\gamma = e/(e-1)$:

$$t = \frac{A(\mathcal{N}_1)}{8} \geq \frac{OPT(\mathcal{N}_1)}{8\gamma} \geq \frac{f(S_1^*)}{8\gamma} \geq \frac{f(S^*)}{32\gamma} \geq \frac{f(S_2^*)}{32\gamma} \geq \frac{f(S_2^*)}{64} \tag{5.10}$$

Hence by lemma 5.2.3 and lemma 5.2.2 we get that it is a $O(1)$ approximation. □

CHAPTER 6

CONCLUSION

Through examples we showed that sequential learning with resource constraints is an area rich in terms of applications. Yet, this area lacked the theoretical tools and there were no general models which captured many of these applications. In this thesis, we have proposed very general models along with algorithms and lower bounds for dealing with such problems. Specifically we have proposed the following three results.

1. We proposed the very general model of Bandits with Knapsacks and developed algorithms with nearly optimal regret guarantees.
2. Then we generalized the results from Bandits with Knapsacks to the contextual setting where we can have side information.
3. We also considered the special case of dynamic procurement and achieved multiplicative approximation when the utility function is submodular.

Above results represent the first and to date most comprehensive understanding of models for sequential learning with resource constraints. In the next section we will outline open problems resulting from our work.

6.1 Open problems

While the work in this thesis resulted in proposing very general models of sequential learning with resource constraints, it also opens up several technical as well as more open ended research programs.

6.1.1 Bandits with Knapsacks

While we prove that the regret bound for PD − BwK is optimal up to logarithmic factors, improved results may be possible for various special cases, especially ones involving discretization over a multi-dimensional action space.

The study of multi-armed bandit problems with large strategy sets has been a very fruitful line of investigation. It seems likely that some of the techniques introduced here could be wedded with the techniques from that literature. In particular, it would be intriguing to try combining our primal-dual algorithm PD − BwK with confidence-ellipsoid algorithms for stochastic linear optimization (e.g. see [19]), or enhancing the Mixture_Elimintation algorithm with the technique of adaptively refined discretization, as in the zooming algortihm of [36].

It is tempting to ask about the *adversarial* version of BwK. However, achieving sublinear regret bounds for such a version appears hopeless even for the fixed-arm benchmark. In order to make progress in the positive direction, one may require a more subtle notion of benchmark, and perhaps also some restrictions on the power of the adversary.

6.1.2 Resourceful contextual bandits

We define a very general setting for contextual bandits with resource constraints. We design an algorithm for this problem, and derive a regret bound which achieves the optimal $\sqrt{T}$ scaling in terms of the time horizon T, and the optimal $\sqrt{\log|\Pi|}$ scaling in terms of the policy set Π. Further, we consider discretization issues, and derive a specific corollary for contextual dynamic pricing

with a single product; we obtain a regret bound that is optimal in the regime $B \geq \Omega(T)$. Finally, we derive a partial lower bound which establishes a stark difference from the non-contextual version.

The main open question is combining provable regret bounds and a computationally efficient implementation. Note that achieving $\sqrt{T}$ regret in a computationally efficient way is a major open question for contextual bandits with policy sets, even without the resource constraints.

Several open questions concern our regret bounds. First, it is desirable to achieve the same regret bounds without assuming a known time horizon T. However, this might not be feasible due to the fact that in RCB, T is essentially one of the resource constraints. Second, our main regret bound is not tight for an arbitrary tuple $(B, T, \texttt{OPT})$; likewise, our corollary for dynamic pricing with a single product is not tight for an arbitrary tuple (B, T). In both cases, both upper and lower bounds can potentially be improved. Third, it is important to work out the discretization issues for more general settings of dynamic pricing/procurement. However, these issues are largely unresolved even for the non-contextual version. Finally, if there are no contexts or resource constraints then one can achieve $O(\log T)$ regret with an instance dependent constant; it is not clear whether one can meaningfully extend this result to settings where either contexts or resource constraints are present.

6.1.3 Non-linear objectives

In our study of procurement markets in chapter 5 we showed that for the special case of dynamic procurement one can get a $O(\log n)$ approximation ratio

when the utility function is a submodular function and $O(1)$ approximation ratio when the utility function is a symmetric submodular function. A very interesting question here is if we can show an algorithm with $O(1)$ approximation for the case of general submodular utility function.

While the above result is for the special case of dynamic procurement we do not have general models of sequential learning under resource constraints with non-trivial guarantees when the utility functions are non-linear. A great open question is to find such general models.

6.1.4 Reactive environments

Our work on sequential learning with resource constraints can be thought of as a mild reactive environment. In a reactive environment a learning algorithm faces the challenge of learning while simultaneously trying to maximize rewards when its action in the current round can affect the future. With resource constraints consuming them in the current round makes them unavailable for the future rounds. But the form of reactivity is mild since we assume the resource consumption is at most $1/B$ fraction of the total budget.

The whole problem of reactive environments seem quite general and there should be lots of applications which can be modelled so. Coming up with good models which can capture many of these applications seems like a very interesting direction.

BIBLIOGRAPHY

[1] Shipra Agrawal and Nikhil R. Devanur. Bandits with concave rewards and convex knapsacks. *CoRR*, abs/1402.5758, 2014.

[2] Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. A dynamic near-optimal algorithm for online linear programming, 2009. Technical report. Available from arXiv at `http://arxiv.org/abs/0911.2974`.

[3] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[4] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002. Preliminary version in *15th ICML*, 1998.

[5] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, 2002. Preliminary version in *36th IEEE FOCS*, 1995.

[6] Moshe Babaioff, Liad Blumrosen, Shaddin Dughmi, and Yaron Singer. Posting prices with unknown distributions. In *ICS*, pages 166–178, 2011.

[7] Moshe Babaioff, Shaddin Dughmi, Robert Kleinberg, and Aleksandrs Slivkins. Dynamic pricing with limited supply. In *13th ACM Conf. on Electronic Commerce (EC)*, 2012.

[8] Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. Matroids, secretary problems, and online mechanisms. In *18th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 434–443, 2007.

[9] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Yaron Singer. Learning on a budget: posted price mechanisms for online procurement. In *13th ACM Conf. on Electronic Commerce (EC)*, pages 128–145, 2012.

[10] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. In *54th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2013.

[11] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. A technical report on `arxiv.org`., May 2013.

[12] Ashwinkumar Badanidiyuru, John Langford, and Aleksandrs Slivkins. Resourceful contextual bandits. *CoRR (To appear at COLT 2014)*, abs/1402.6779, 2014.

[13] Omar Besbes and Assaf Zeevi. Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research*, 57:1407–1420, 2009.

[14] Omar Besbes and Assaf J. Zeevi. Blind network revenue management. *Operations Research*, 60(6):1537–1550, 2012.

[15] Avrim Blum, Vijay Kumar, Atri Rudra, and Felix Wu. Online learning in online auctions. In *14th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 202–204, 2003.

[16] Sébastien Bubeck and Nicolo Cesa-Bianchi. Regret Analysis of Stochastic and Nonstochastic Multi-armed Bandit Problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.

[17] Sébastien Bubeck and Aleksandrs Slivkins. The best of both worlds:

stochastic and adversarial bandits. In *25th Conf. on Learning Theory (COLT)*, 2012.

[18] Nicoló Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. Regret minimization for reserve prices in second-price auctions. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, 2013.

[19] Varsha Dani, Thomas P. Hayes, and Sham Kakade. Stochastic Linear Optimization under Bandit Feedback. In *21th Conf. on Learning Theory (COLT)*, pages 355–366, 2008.

[20] Nikhil R. Devanur and Thomas P. Hayes. The AdWords problem: Online keyword matching with budgeted bidders under random permutations. In *Proceedings 10th ACM Conference on Electronic Commerce (EC-2009)*, pages 71–78, 2009.

[21] Nikhil R. Devanur, Kamal Jain, Balasubramanian Sivan, and Christopher A. Wilkens. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *Proceedings 12th ACM Conference on Electronic Commerce (EC-2011)*, pages 29–38, 2011.

[22] Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

[23] Miroslav Dudik, Daniel Hsu, Satyen Kale, Nikos Karampatziakis, John Langford, Lev Reyzin, and Tong Zhang. Efficient optimal leanring for contextual bandits. In *27th Conf. on Uncertainty in Artificial Intelligence (UAI)*, 2011.

[24] E. B. Dynkin. The optimum choice of the instant for stopping a Markov process. *Soviet Math. Dokl*, 4, 1963.

[25] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[26] Jon Feldman, Monika Henzinger, Nitish Korula, Vahab S. Mirrokni, and Clifford Stein. Online stochastic packing applied to display ad allocation. In *Proc. 18th European Symposium on Algorithms (ESA)*, pages 182–194, 2010.

[27] D. A. Freedman. On tail probabilities for martingales. *The Annals of Probability*, 3:100–118, 1975.

[28] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[29] Naveen Garg and Jochen Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM J. Computing*, 37(2):630–652, 2007.

[30] Sudipto Guha and Kamesh Munagala. Multi-armed bandits with metric switching costs. In *Proc. 36th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 496–507, 2009.

[31] Anupam Gupta, Ravishankar Krishnaswamy, Marco Molinaro, and R. Ravi. Approximation algorithms for correlated knapsacks and non-martingale bandits. In *52nd IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 827–836, 2011.

[32] Elad Hazan and Nimrod Megiddo. Online Learning with Prior Information. In *20th Conf. on Learning Theory (COLT)*, pages 499–513, 2007.

[33] Samir Khuller, Anna Moss, and Joseph (Seffi) Naor. The budgeted maximum coverage problem. *Inf. Process. Lett.*, 70(1):39–45, 1999.

[34] Robert Kleinberg. Nearly tight bounds for the continuum-armed bandit problem. In *18th Advances in Neural Information Processing Systems (NIPS)*, 2004.

[35] Robert Kleinberg and Tom Leighton. The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In *44th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 594–605, 2003.

[36] Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-Armed Bandits in Metric Spaces. In *40th ACM Symp. on Theory of Computing (STOC)*, pages 681–690, 2008.

[37] Robert D. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *SODA*, pages 630–631, 2005.

[38] Robert D. Kleinberg and Frank T. Leighton. The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, 2003.

[39] T. L. Lai and Herbert Robbins. Asymptotically efficient adaptive allocations rules. *Adv. in Appl. Math.*, 6:4–22, 1985.

[40] John Langford and Tong Zhang. The Epoch-Greedy Algorithm for Contextual Multi-armed Bandits. In *21st Advances in Neural Information Processing Systems (NIPS)*, 2007.

[41] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–260, 1994.

[42] Tyler Lu, Dávid Pál, and Martin Pál. Showing Relevant Ads via Lipschitz Context Multi-Armed Bandits. In *14thIntl. Conf. on Artificial Intelligence and Statistics (AISTATS)*, 2010.

[43] Marco Molinaro and R. Ravi. Geometry of online packing linear programs. In *Proc. 39th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 701–713, 2012.

[44] Sandeep Pandey, Deepak Agarwal, Deepayan Chakrabarti, and Vanja Josifovski. Bandits for Taxonomies: A Model-based Approach. In *SIAM Intl. Conf. on Data Mining (SDM)*, 2007.

[45] Sandeep Pandey, Deepayan Chakrabarti, and Deepak Agarwal. Multi-armed Bandit Problems with Dependent Arms. In *24th Intl. Conf. on Machine Learning (ICML)*, 2007.

[46] Serge A. Plotkin, David B. Shmoys, and Eva Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20:257–301, 1995.

[47] Adish Singla and Andreas Krause. Truthful incentives in crowdsourcing tasks using regret minimization mechanisms. In *22nd Intl. World Wide Web Conf. (WWW)*, pages 1167–1178, 2013.

[48] Aleksandrs Slivkins and Jennifer Wortman Vaughan. Online decision making in crowdsourcing markets: Theoretical challenges. *SIGecom Exchanges*, 12(2), December 2013. Position Paper and survey.

[49] Maxim Sviridenko. A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.*, 32(1):41–43, 2004.

[50] William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):pp. 285–294, 1933.

[51] Long Tran-Thanh. *Budget-Limited Multi-Armed Bandits*. PhD thesis, University of Southampton, 2012.

[52] Long Tran-Thanh, Archie Chapman, Enrique Munoz de Cote, Alex Rogers, and Nicholas R. Jennings. ϵ-first policies for budget-limited multi-armed bandits. In *Proc. Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)*, pages 1211–1216, 2010.

[53] Long Tran-Thanh, Archie Chapman, Alex Rogers, and Nicholas R. Jennings. Knapsack based optimal policies for budget-limited multi-armed bandits. In *Proc. Twenty-Sixth AAAI Conference on Artificial Intelligence (AAAI-12)*, pages 1134–1140, 2012.

[54] Peter Whittle. Multi-armed bandits and the Gittins index. *J. Royal Statistical Society, Series B*, 42(2):143–149, 1980.